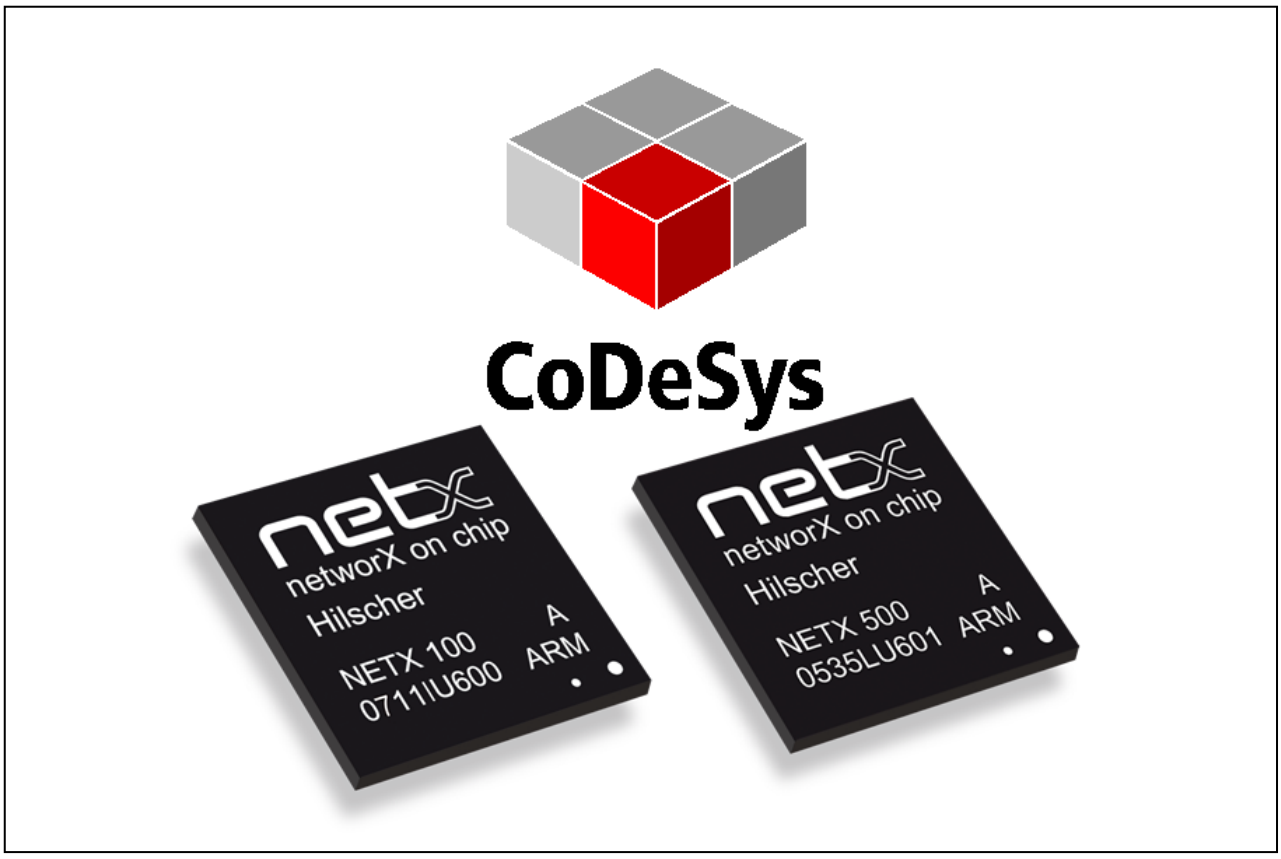
User Manual

netPLC with CoDeSys

Commissioning

V3.5



Hilscher Gesellschaft für Systemautomation mbH

www.hilscher.com

DOC120113UM02EN | Revision 2 | English | 2014-01 | Released | Public

Table of Contents

1 INTRODUCTION 5
1.1 About the Manual 5
1.1.1 List of Revisions 5
1.1.2 Conventions in this Manual 6
1.2 netPLC with CoDeSys 7
1.2.1 Reference to Hardware, Firmware, Server, Driver and Software 7
1.2.2 Documentation netPLC 8
1.3 Legal Notes 9
1.3.1 Copyright 9
1.3.2 Important Notes 9
1.3.3 Exclusion of Liability 10
1.3.4 Warranty 10
1.3.5 Export Regulations 11
1.3.6 Registered Trademarks 11
1.4 Licenses 11

2 SAFETY 12
2.1 Safety Instructions 12
2.2 Intended Use 12
2.3 Personnel Qualification 12
2.4 Obligation to Read and Understand the Manual 12

3 DESCRIPTION AND REQUIREMENTS 13
3.1 Description 13
3.2 System Requirements 13
3.2.1 System Requirements PC 13
3.2.2 System Requirements CoDeSys 13

4 CREATE FIRST PROJECT FOR NETPLC COMPONENT (EXAMPLE NPLC-C100) 14
4.1 Overview 14
4.2 Preparation 15
4.2.1 netPLC CoDeSys PC Card Server 15
4.2.2 Start CoDeSys 15
4.3 Create a PLC Program, Selecting Device and Programming Mode 17
4.3.1 Add Host I/Os to NPLC-C100 Slot PLC 19
4.3.2 Add Local I/Os to NPLC-M100 Module 22
4.4 Create Bus Configuration 25
4.4.1 Select Fieldbus System for the Controller 25
4.4.2 Configure PROFIBUS Master 27
4.4.3 Add CANopen Master Configuration Window 29
4.4.4 Configure DeviceNet Scanner 33
4.5 Map Bus Structure 37
4.5.1 Import Slave Device Files 37
4.5.2 Add and configure Slave Device 38
4.5.3 Identify I/O Addresses of the Slave 40

© Hilscher, 2009-2014

4.6 Create PLC Program 41
4.6.1 Create Symbol Configuration 44
4.6.2 Build Project 47
4.7 Connect to the PLC 48
4.7.1 Define Gateway 48
4.7.2 Scan Network and Set Active Path 49
4.8 Download PLC Program and Bus Configuration 51
4.8.1 Login and Download 51
4.8.2 Create Boot Application 53
4.8.3 Remove Boot Application 53
4.8.4 Start PLC Program 54
4.9 Disconnect from the PLC 55

5 VISUALIZING 56
5.1 OPC Configurator 57
5.2 OPC Client 61

6 HOST I/O INTERFACE 62

7 FUNCTIONS 63
7.1 Add Functions Library 63
7.2 Functions – HilPlcSysFunctions 64
7.2.1 Battery Status – StatusOfBattery 66
7.2.2 Status External Voltage Supply – StatusOfExt24VDC 68
7.2.3 Status PCI Power Supply – StatusOfPCIPower 69
7.3 Reset Behavior 70
7.4 RUN/STOP Switch 70
7.5 Monitor netPLC C100/M100 Battery 71

8 BUS DIAGNOSTIC 72
8.1 Start Up and Test the Bus 72
8.1.1 Login and Download Program 72
8.1.2 Device Assignment 73
8.1.3 Verify Input and Output Data 74

9 TROUBLESHOOTING 75
9.1 Error Message During Boot Application Creation 75

10 LEDS 76
10.1 LED SYS 76
10.2 LED APL 77
10.3 LED CANopen Master 77
10.4 LED DeviceNet Master 77
10.5 LED PROFIBUS DP-Master 78

11 ERROR CODES 79
11.1 Error Codes Functions HilPlcSysFunctions 79

12 TECHNICAL DATA PROTOCOLS 80
12.1 CANopen Master and DeviceNet Master 80

© Hilscher, 2009-2014

12.2 PROFIBUS DP-Master 82

13 APPENDIX 83
13.1 netPLC CoDeSys Server Program 83
13.1.1 Starting netPLC CoDeSys Server 83
13.1.2 netPLC CoDeSys Server Window 84
13.1.3 Stopping netPLC CoDeSys Server 85
13.2 Example Projects 86
13.2.1 netPLC System Function 86
13.2.2 netPLC Open Process Control (OPC) 87
13.2.3 netPLC Real Time Clock 87
13.2.4 NPLC-C100-DP PROFIBUS Simple IO 88
13.2.5 NPLC-C100-DP with Host IOs 88
13.3 Information on Utilized Capacity 89
13.4 List of Figures 91
13.5 List of Tables 92
13.6 Contacts 93

© Hilscher, 2009-2014

1 Introduction

1.1 About the Manual

This user manual describes the commissioning of netPLC components with the PLC programming software CoDeSys.

netPLC components are

- Slot PLC NPLC-C100 and
- Module PLC NPLC-M100.

1.1.1 List of Revisions

Index	Date	Chapter	Revisions
1	2012-05-21	all	created
2	2014-01-27	1.2.1	Section *Reference to Hardware, Firmware, Server, Driver and Software* updated.
		4	Chapter *Create first Project for netPLC Component (Example NPLC-C100)* updated.
		4.3.1	Section *Add Host I/Os to NPLC-C100 Slot PLC* added.
		6	Chapter *Host I/O Interface* added.
		12.1	Technical data for CANopen Master and DeviceNet Master together in one section and updated.
		12.2	Technical data for *PROFIBUS DP-Master* updated and DPV1 functions added.
		13.2.4	Section about example *NPLC-C100-DP PROFIBUS Simple IO* added.
		13.2.5	Section about example *NPLC-C100-DP with Host IOs* added.

Table 1: List of Revisions

© Hilscher, 2009-2014

1.1.2 Conventions in this Manual

Operation instructions, a result of an operation step or notes are marked as follows:

Operation Instructions:

➢ <instruction>

or

1. <instruction>
2. <instruction>

Results:

➪ <result>

Notes:

Important: <important note>

Note: <note>

<note, where to find further information>

Positions in Figures

The *Positions* ①, ②, ③ **...** or ⓐ, ⓑ, ⓒ **...** or Ⓐ, Ⓑ, Ⓒ **...** refer to the figure used in that section. If the numbers reference to a section outside the current section then a cross reference to that section and figure is indicated.

© Hilscher, 2009-2014

1.2 netPLC with CoDeSys

1.2.1 Reference to Hardware, Firmware, Server, Driver and Software

Note: The listed hardware revision, firmware and driver versions or versions of the programming software CoDeSys functionally belong together.

netPLC Components (Hardware)

netPLC components	Order number	Revision
NPLC-C100-DP/CDS-OPC	1800.410/CDS-OPC	4
NPLC-C100-CO/CDS-OPC	1800.500/CDS-OPC	4
NPLC-C100-DN/CDS-OPC	1800.510/CDS-OPC	4
NPLC-M100-DP/CDS-OPC	1830.410/CDS-OPC	3

Table 2: Reference to netPLC Components (Hardware)

Firmware

Firmware File	Device Type	Fieldbus System	Firmware Version
SC111000.NXF	NPLC-C100-DP	PROFIBUS DP Master	4.0
SC116000.NXF	NPLC-C100-CO	CANopen Master	4.0
	NPLC-C100-DN	DeviceNet Master	4.0
SM211000.NXF	NPLC-M100-DP	PROFIBUS DP Master	4.0

Table 3: Reference to Firmware

Driver

Driver	Driver version
cifX Device Driver	1.1.x.x
USB Driver	5.1.2600.2180

Table 4: Reference to Driver

Server and TCP/IP Gateway

Software	Software Version
CoDeSys Gateway	3.5.0.0
CoDeSys Service Control	3.5.0.0
CoDeSys Control Win V3	3.5.0.0

Table 5: Reference to Server

Software

Software	Software version
CoDeSys	V3.5

Table 6: Reference to Software

© Hilscher, 2009-2014

1.2.2 Documentation netPLC

The following documentation overview gives information about where to find further information and refers the corresponding manual.

All manuals listed in the overview below can be found in the **Documentation** directory on the CD delivered, in the Adobe Acrobat® Reader format (PDF).

Manual	Title Contents	File Name
User manual	netPLC with CoDeSys, software installation Contents of the product DVD Requirements System overview Installing software Technical data	netPLC with CoDeSys – Software Installation UM xx EN.pdf (This document)
User manual	Slot PLC NPLC-C100, hardware installation Requirements Device drawings Installing hardware (slot PLC) Changing battery Hardware description Technical data hardware Remark: The description of the LED is part of the document ‘netPLC with CoDeSys, Commissioning’.	Slot PLC NPLC-C100 - Hardware Installation UM xx EN.pdf
User manual	netPLC with CoDeSys, Commissioning Create first project for netPLC Creating a PLC program Create bus configuration Connect to PLC Download PLC program and bus configuration Visualizing Functions Bus diagnostic Troubleshooting Description of LEDs	netPLC with CoDeSys - Commissioning UM xx EN.PDF
User documentation	CoDeSys V3, Installation and first steps	CoDesys Installation and Start.pdf
User documentation	OPC Server 3, Installtion and Usage	OPC_V3_how_to_use_E.pdf
User manual	Wiring instructions, PROFIBUS, CANopen, DeviceNet, AS-Interface, CompoNet, CC-Link That document contains information about cable characteristics, max. cable length in dependence of the baudrate as well as termination resistors.	Wiring Instructions UM xx EN.pdf

Table 7: Documentations netPLC CoDeSys

Note: After having installed the CoDeSys programming system, you will find additional documents in the installation directory, usually under **C:\Programs\3S CoDeSys\CoDeSys\Documentation**

© Hilscher, 2009-2014

1.3 Legal Notes

1.3.1 Copyright

© Hilscher, 2009-2014, Hilscher Gesellschaft für Systemautomation mbH

All rights reserved.

The images, photographs and texts in the accompanying material (user manual, accompanying texts, documentation, etc.) are protected by German and international copyright law as well as international trade and protection provisions. You are not authorized to duplicate these in whole or in part using technical or mechanical methods (printing, photocopying or other methods), to manipulate or transfer using electronic systems without prior written consent. You are not permitted to make changes to copyright notices, markings, trademarks or ownership declarations. The included diagrams do not take the patent situation into account. The company names and product descriptions included in this document may be trademarks or brands of the respective owners and may be trademarked or patented. Any form of further use requires the explicit consent of the respective rights owner.

1.3.2 Important Notes

The user manual, accompanying texts and the documentation were created for the use of the products by qualified experts, however, errors cannot be ruled out. For this reason, no guarantee can be made and neither juristic responsibility for erroneous information nor any liability can be assumed. Descriptions, accompanying texts and documentation included in the user manual do not present a guarantee nor any information about proper use as stipulated in the contract or a warranted feature. It cannot be ruled out that the user manual, the accompanying texts and the documentation do not correspond exactly to the described features, standards or other data of the delivered product. No warranty or guarantee regarding the correctness or accuracy of the information is assumed.

We reserve the right to change our products and their specification as well as related user manuals, accompanying texts and documentation at all times and without advance notice, without obligation to report the change. Changes will be included in future manuals and do not constitute any obligations. There is no entitlement to revisions of delivered documents. The manual delivered with the product applies.

Hilscher Gesellschaft für Systemautomation mbH is not liable under any circumstances for direct, indirect, incidental or follow-on damage or loss of earnings resulting from the use of the information contained in this publication.

1.3.3 Exclusion of Liability

The software was produced and tested with utmost care by Hilscher Gesellschaft für Systemautomation mbH and is made available as is. No warranty can be assumed for the performance and flawlessness of the software for all usage conditions and cases and for the results produced when utilized by the user. Liability for any damages that may result from the use of the hardware or software or related documents, is limited to cases of intent or grossly negligent violation of significant contractual obligations. Indemnity claims for the violation of significant contractual obligations are limited to damages that are foreseeable and typical for this type of contract.

It is strictly prohibited to use the software in the following areas:

- for military purposes or in weapon systems;
- for the design, construction, maintenance or operation of nuclear facilities;
- in air traffic control systems, air traffic or air traffic communication systems;
- in life support systems;
- in systems in which failures in the software could lead to personal injury or injuries leading to death.

We inform you that the software was not developed for use in dangerous environments requiring fail-proof control mechanisms. Use of the software in such an environment occurs at your own risk. No liability is assumed for damages or losses due to unauthorized use.

1.3.4 Warranty

Although the hardware and software was developed with utmost care and tested intensively, Hilscher Gesellschaft für Systemautomation mbH does not guarantee its suitability for any purpose not confirmed in writing. It cannot be guaranteed that the hardware and software will meet your requirements, that the use of the software operates without interruption and that the software is free of errors. No guarantee is made regarding infringements, violations of patents, rights of ownership or the freedom from interference by third parties. No additional guarantees or assurances are made regarding marketability, freedom of defect of title, integration or usability for certain purposes unless they are required in accordance with the law and cannot be limited. Warranty claims are limited to the right to claim rectification.

© Hilscher, 2009-2014

1.3.5 Export Regulations

The delivered product (including the technical data) is subject to export or import laws as well as the associated regulations of different counters, in particular those of Germany and the USA. The software may not be exported to countries where this is prohibited by the United States Export Administration Act and its additional provisions. You are obligated to comply with the regulations at your personal responsibility. We wish to inform you that you may require permission from state authorities to export, re-export or import the product.

1.3.6 Registered Trademarks

Windows® XP, Windows® 7 are registered trademarks of Microsoft Corporation

Adobe-Acrobat® is a registered trademark of the Adobe Systems Incorporated.

1.4 Licenses

Operating the netPLC component with CoDeSys PLC function requires a license. This license is embedded in the netPLC device hardware and is included in the scope of delivery. It allows the use and operation of the CoDeSys programming system and of the PLC runtime environment in the hardware.

A license can not be deleted in the netPLC hardware nor converted.

For operating as fieldbus master, a master license is embedded in the netPLC hardware. This license allows operation as master in the corresponding fieldbus system.

© Hilscher, 2009-2014

2 Safety

2.1 Safety Instructions

The user manual, the accompanying texts and the documentation are written for the use of the products by educated personnel. When using the products, all Safety Messages, Safety Messages, Property Damage Messages and all valid legal regulations have to be obeyed. Technical knowledge is presumed. The user has to assure that all legal regulations are obeyed.

2.2 Intended Use

The netPLC components described in this user manual have PLC functionality and fieldbus communication.

netPLC components are

- Slot PLC NPLC-C100 and
- Module PLC NPLC-M100.

Depending on the used netPLC component the fieldbus systems listed in section *Reference to Hardware, Firmware, Server, Driver and Software* (page 7) can be automated.

2.3 Personnel Qualification

The netPLC component must only be installed, configured and removed by qualified personnel.

2.4 Obligation to Read and Understand the Manual

Important! Read and understand all instructions in this manual before installation or use of your device to avoid injury.

© Hilscher, 2009-2014

3 Description and Requirements

3.1 Description

CoDeSys is a PLC programming software. Matching the IEC 61131-3 standard it supports all standard programming languages. This user manual describes the device dependant programming for netPLC components, including the description of the functions implemented in CoDeSys for monitoring netPLC components.

netPLC components are

- Slot PLC NPLC-C100 and
- Module PLC NPLC-M100.

3.2 System Requirements

3.2.1 System Requirements PC

- PC with min. 1,8 GHz processor recommended 3,0 GHz.
- Windows® XP or Windows® 7
- DVD ROM drive
- Graphic resolution: min. 1024 x 768 pixel or higher
- Keyboard and Mouse
- For slot PLC NPLC-C100: PC with PCI interface with 3.3 V respectively 5 V for hardware revision 4 of the NPLC-C100

3.2.2 System Requirements CoDeSys

System requirements for the programming system CoDeSys

- PC with min. 1,8 GHz processor recommended 3,0 GHz.
- Windows® XP or Windows® 7
- .NET Framework 4.0 (will be installed together with CoDeSys, if not already installed. Access to the internet is required.)
- Free disk space: min. 600 MByte, recommended 1 GByte
- RAM: min. 512 MByte, recommended 1024 MByte
- Graphic resolution: min. 1024 x 768 pixel
- Keyboard and Mouse

© Hilscher, 2009-2014

4 Create first Project for netPLC Component (Example NPLC-C100)

4.1 Overview

This chapter describes how to use the CoDeSys programming system to create a project for a netPLC component.

The following example features a NPLC-C100-DP. Prerequisite for operation is the presence of at least one PROFIBUS DP slave and a PROFIBUS cable.

The subsequently described procedures are in general also practicable for other netPLC components. Possible deviations from the given example which would occur if other netPLC components were used, are pointed out in the corresponding section.

If the CoDeSys project is to be loaded to the netPLC component via USB interface, an A to Mini B type USB cable is required.

© Hilscher, 2009-2014

4.2 Preparation

4.2.1 netPLC CoDeSys PC Card Server

The netPLC CoDeSys Server is only needed if communication with a slot PLC NPLC-C100 takes place via the PCI bus of a PC. Information on when the server is needed, can be inferred from the figures in the *System Overview* section of the user manual *netPLC with CoDeSys, Software Installation*.

The netPLC CoDeSys PC card server program is installed to start automatically with each start of the PC.

Note: Make sure that the server runs, if communication via the PCI bus of the PC is used to communicate to the slot PLC NPLC-C100.

Check, if netPLC CoDeSys server is running.

- Check, if the netPLC CoDeSys server is running. This is indicated by the symbol in the system tray.
- If the netPLC CoDeSys server is not running, start it by choosing **Start > All Programs > Hilscher GmbH > netPLC CoDeSys Server**.

→ The system tray shows the symbol .

4.2.2 Start CoDeSys

Start CoDeSys

- Select **Start > All Programs > 3S CoDeSys > CoDeSys > CoDeSys V3.5**

or

- Start CoDeSys by a double click on the desktop icon

→ CoDeSys is started.

© Hilscher, 2009-2014

➪ The CoDeSys GUI appears.

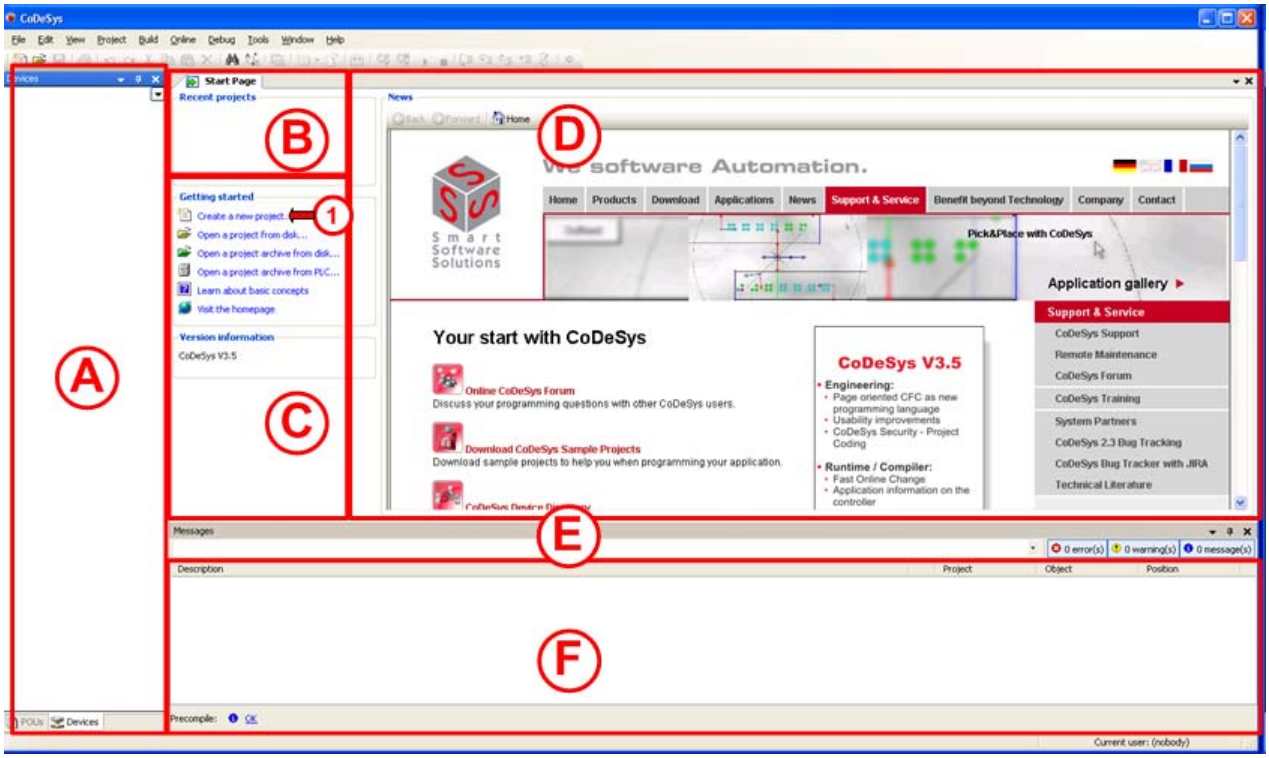


Figure 1: CoDeSys Main Window

The main window is divided into the following areas:

Ⓐ Shows objects belonging to a project, like used **Devices** or **POUs** (Program Organization Units)

Areas Ⓑ, Ⓒ and Ⓓ form the actual **Start Page**. It consists of:

Ⓑ Area showing recently used objects

Ⓒ Area featuring links for starting a project

Ⓓ HTML Browser featuring links to 3S CoDeSys homepage

Ⓔ Messages area

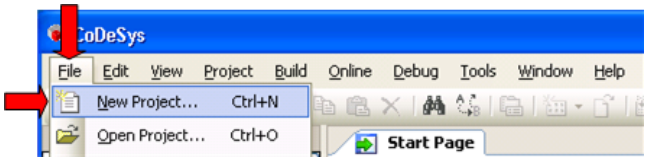
Ⓕ Description area

© Hilscher, 2009-2014

4.3 Create a PLC Program, Selecting Device and Programming Mode

1. Create a new project.

➢ Select **File > New Project**.



or use **Create a new project** link ① in area Ⓒ (as indicated in Figure 1).

➢ The **New Project** window opens.

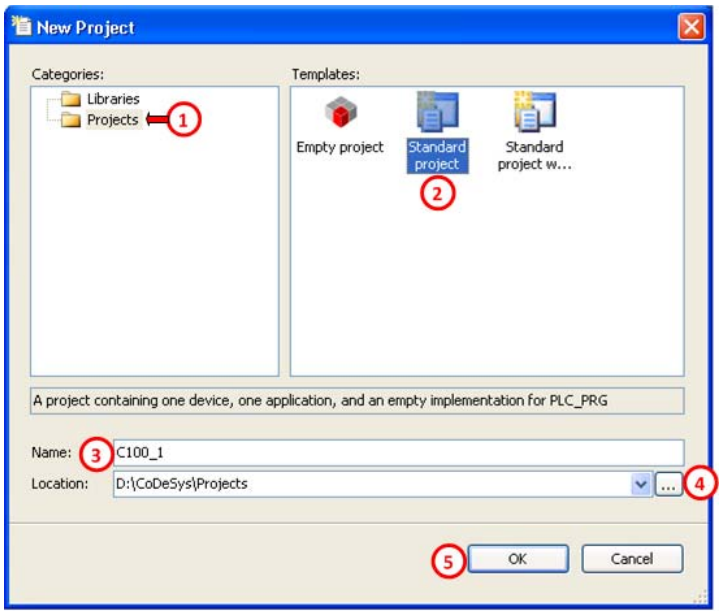


Figure 2: New Project

2. Set up project-relevant information.

➢ Select **Categories > Projects** ①.

➢ Select **Templates > Standard project** ②.

➢ In the **Name** field ③, fill in the project name e. g. **FirstProject**.

➢ In the **Location** field select a folder to save the project data, e. g. **D:\CoDeSys\Projects**.

Use the browse button ④ to open the **Browse For Folder** dialog, in which you can select an existing folder or create a new folder.

➢ Click **OK** ⑤.

© Hilscher, 2009-2014

➧ The **New Project** window is closed and the **Standard Project** selection window appears.

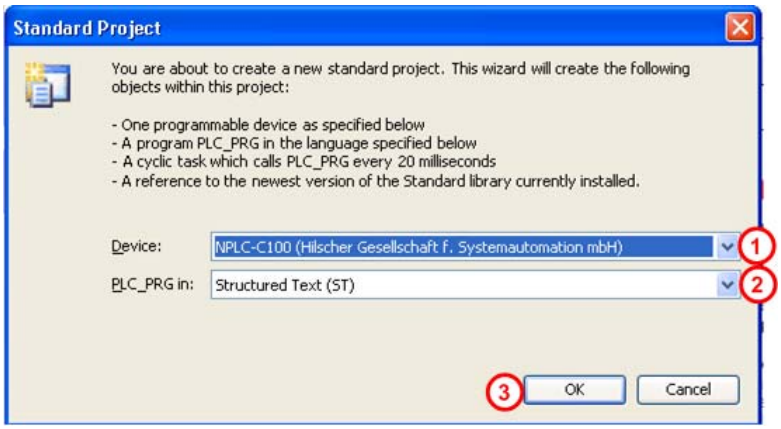

Standard Project
You are about to create a new standard project. This wizard will create the following objects within this project:
- One programmable device as specified below
- A program PLC_PRG in the language specified below
- A cyclic task which calls PLC_PRG every 20 milliseconds
- A reference to the newest version of the Standard library currently installed.
Device:
NPLC-C100 (Hilscher Gesellschaft f. Systemautomation mbH)
PLC_PRG in:
Structured Text (ST)
1
2
3
OK
Cancel

Figure 3: Standard Project, Device Selection

3. Specify Standard Project.

➢ In the **Device** field ① select the netPLC device which you want to use in your project (see Figure 3). This example uses the **NPLC-C100 (Hilscher Gesellschaft f. Systemautomation mbH)**.

➢ In the **PLC_PRG in** field ② select the programming type **Structured Text (ST)**.

➢ Click **OK** ③.

➧ The **Standard Project** window closes and the device tree is displayed in the **Object view** area Ⓐ of the main window.

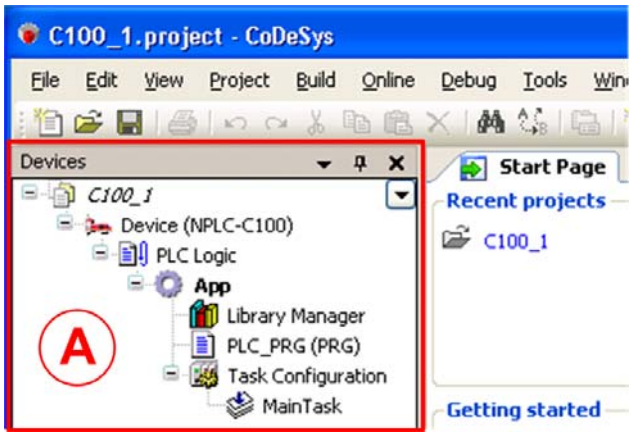


Figure 4: Standard Project, Device Overview

4. Switch to **Devices** view.

➢ Check whether the objects view Ⓐ is set to **Devices**.

➢ If the objects view is set to **POUs** view, click on the Devices tab located in the bottom of the objects view.

➧ The **Devices** view is active.

© Hilscher, 2009-2014

5. Save the project.

➢ Select **File > Save Project** to save the project.

➧ The project is saved.

4.3.1 Add Host I/Os to NPLC-C100 Slot PLC

If you are using a NPLC-C100 slot PLC, you can access inputs and outputs of the PLC from the host. A Windows application program on the host can access these I/O.

The inputs are located at addresses %IB0 to %IB511 and the outputs are located at addresses %QB0 to %QB511, as long as slave devices are added behind the **Host_I_O_Interface**.

In order to use host I/O in the control program, you have to add them to the project. To do so, proceed as follows:

Add host inputs/outputs.

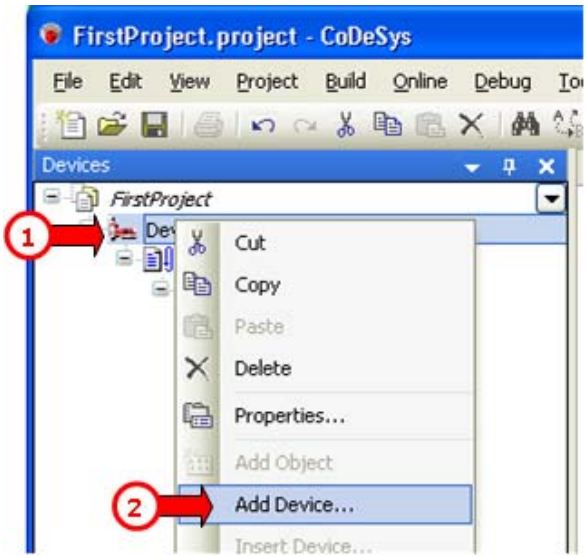


Figure 5: NPLC-C100 Add Host I/O (1)

➢ In the **Devices** tree right click on **Device (NPLC-C100)** ① to open the context menu.

➢ Select **Add Device…** ② in the context menu.

➧ The **Add Device** window opens.

© Hilscher, 2009-2014

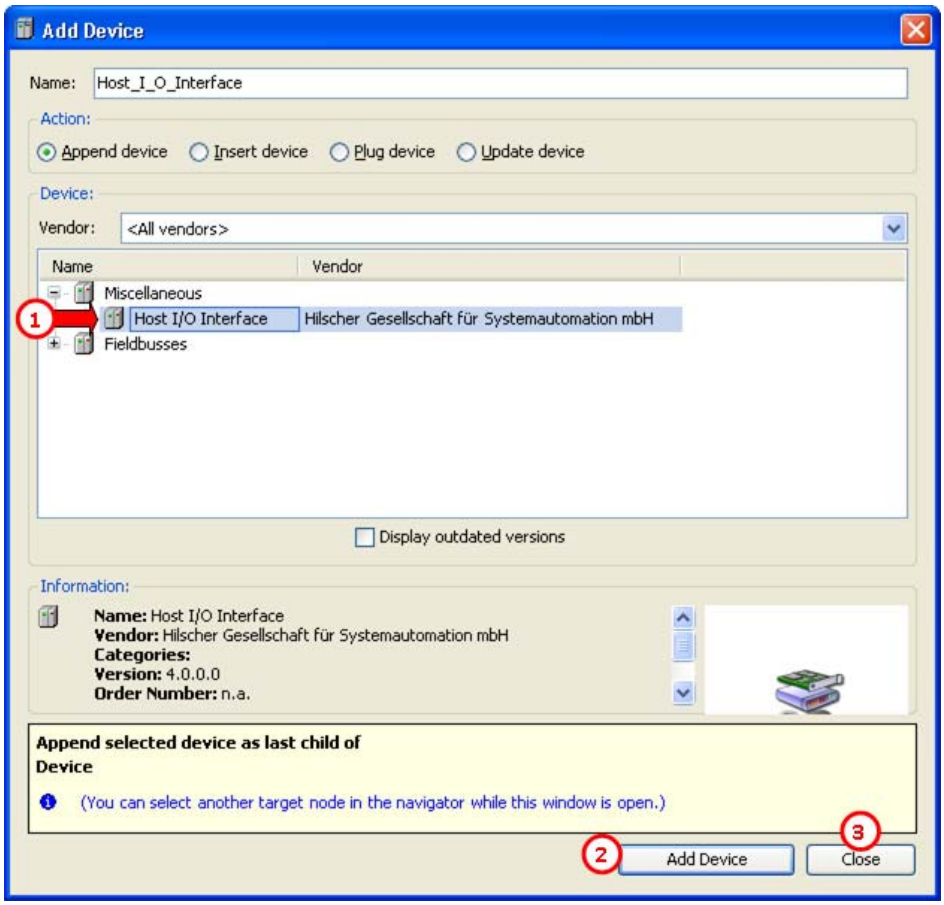


Figure 6: NPLC-C100 Add Host I/Os (2)

- In the **Device** list select **Host I/O Interface** ① (located below **Miscellaneous**).
- Click **Add Device** ②.
- Click **Close** ③.

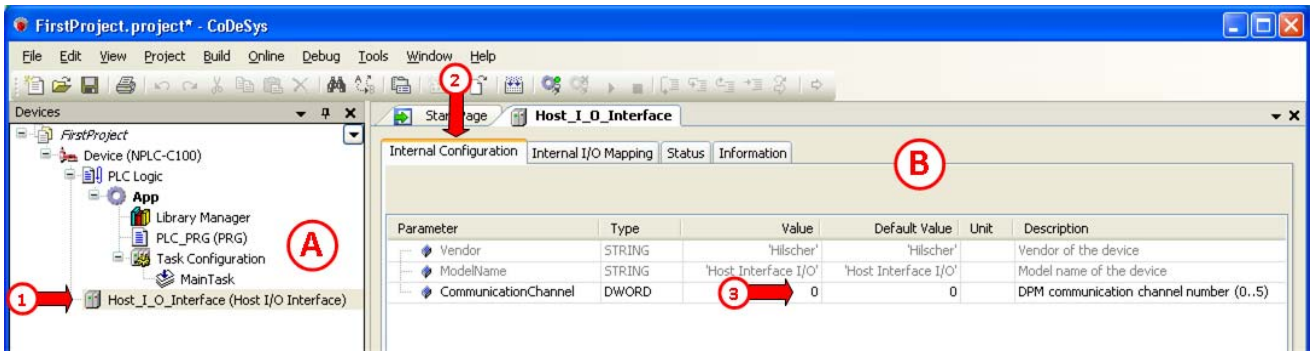


Figure 7: C100 Configure Host I/Os (1)

- In the **Devices** tree Ⓐ item **Host_I_O_Interface** ① has been added.
- In the **Devices** tree double click **Host_I_O_Interface** ①.
- Select **Internal Configuration** tab ②.
- Use for the **CommunicationChannel** ③ the default value 0.

© Hilscher, 2009-2014

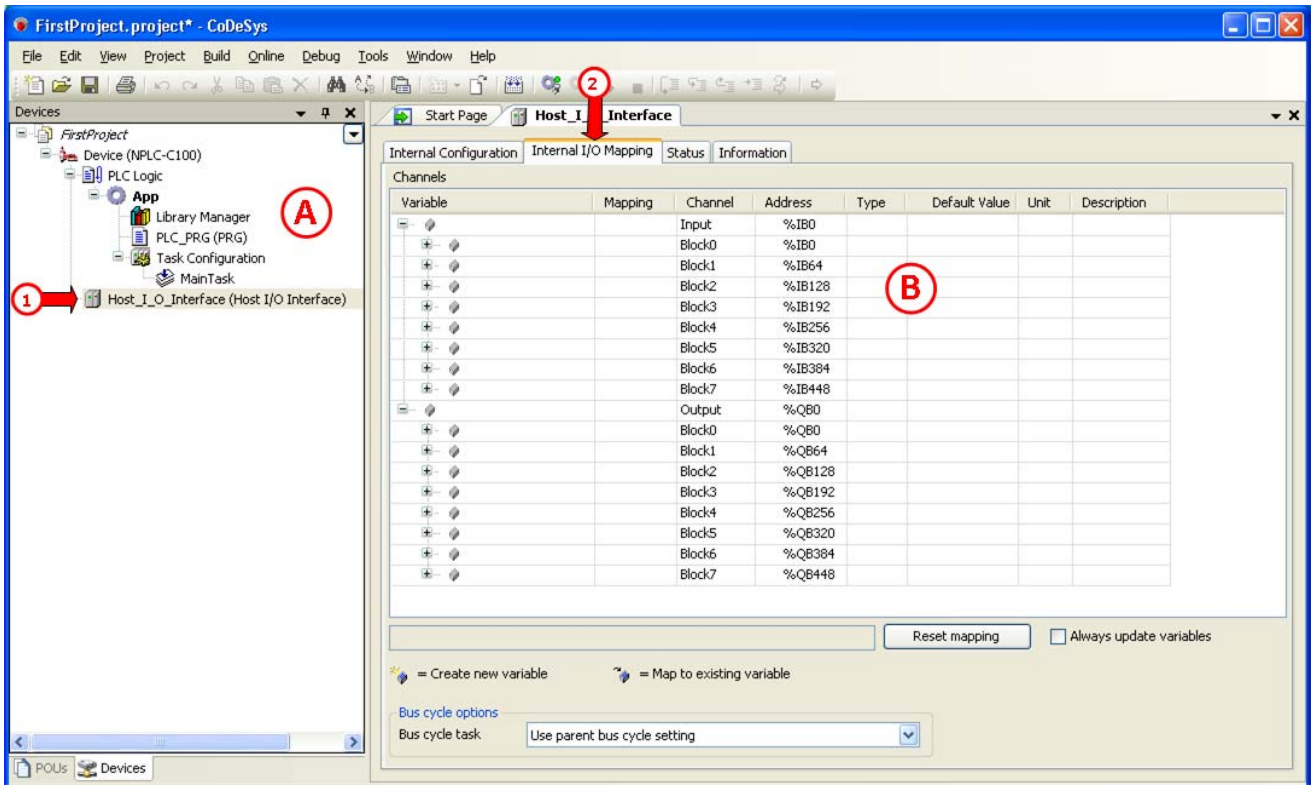


Figure 8: C100 Configure Host I/Os (2)

- If not already selected, double click **Host_I_O_Interface** ① in the **Devices** tree.
- Select **Internal I/O Mapping** tab ②.
- Now you can double click individual fields in the table, in order to configure signals for bits or words Ⓑ.

© Hilscher, 2009-2014

4.3.2 Add Local I/Os to NPLC-M100 Module

If you are using a NPLC-M100 module PLC, you can use local inputs and outputs. In order to use these in the control program, you have to make them known to the project. To do so, proceed as follows:

Add local inputs/outputs

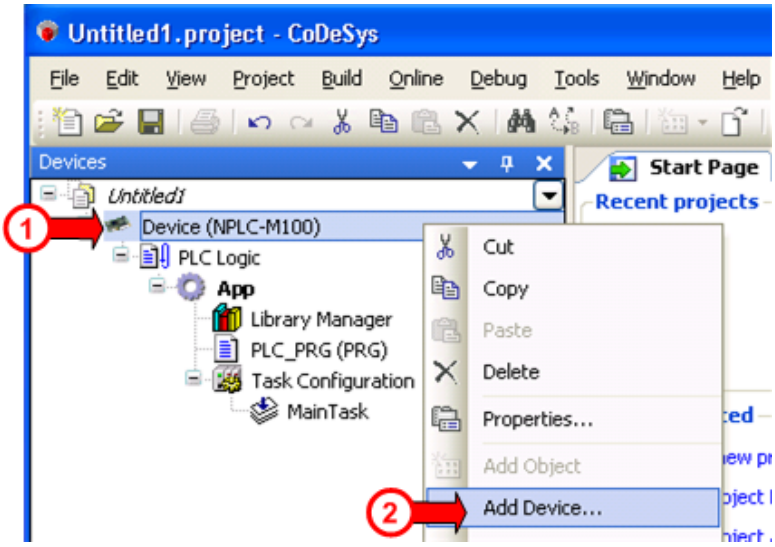


Figure 9: NPLC-M100 Add Local I/O (1)

- In the **Devices** tree right click on **Device (NPLC-M100)** (1) to open context menu.
- Select **Add Device…** (2) in the context menu.
- The **Add Device** window opens.

© Hilscher, 2009-2014

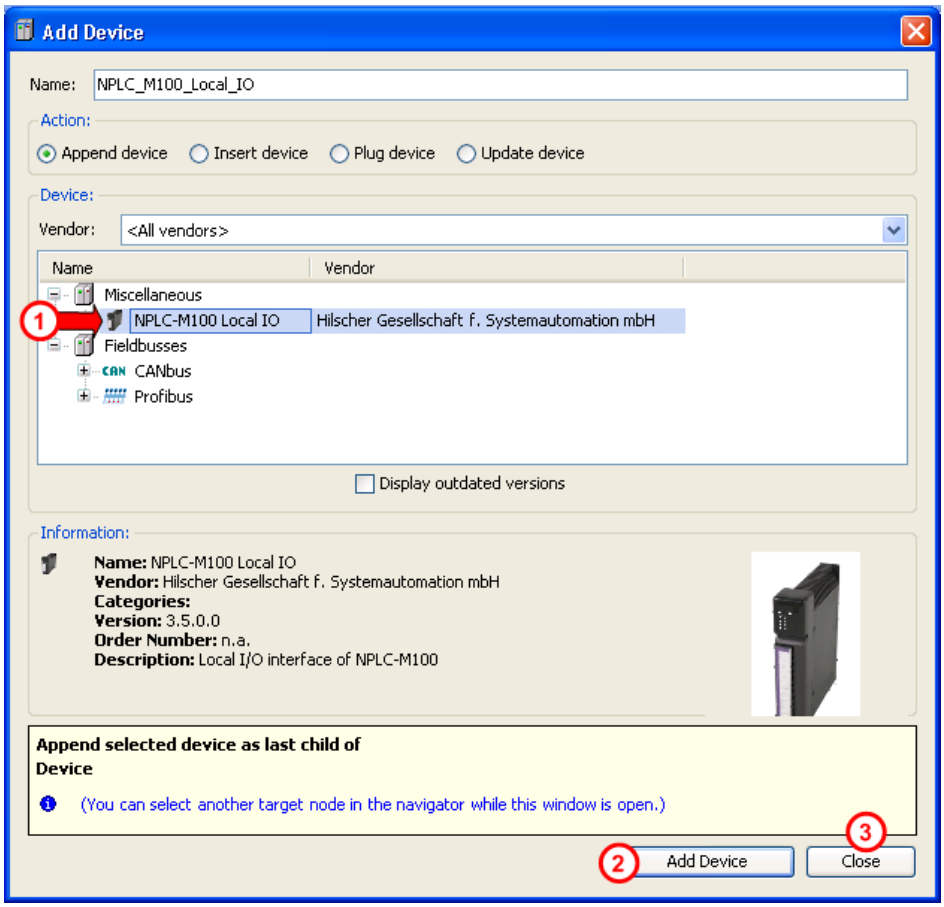


Figure 10: NPLC-M100 Add Local I/Os (2)

- In the **Device** list select **NPLC-M100 Local IO** (1) (located below **Miscellaneous**).
- Click **Add Device** (2).
- Click **Close** (3).

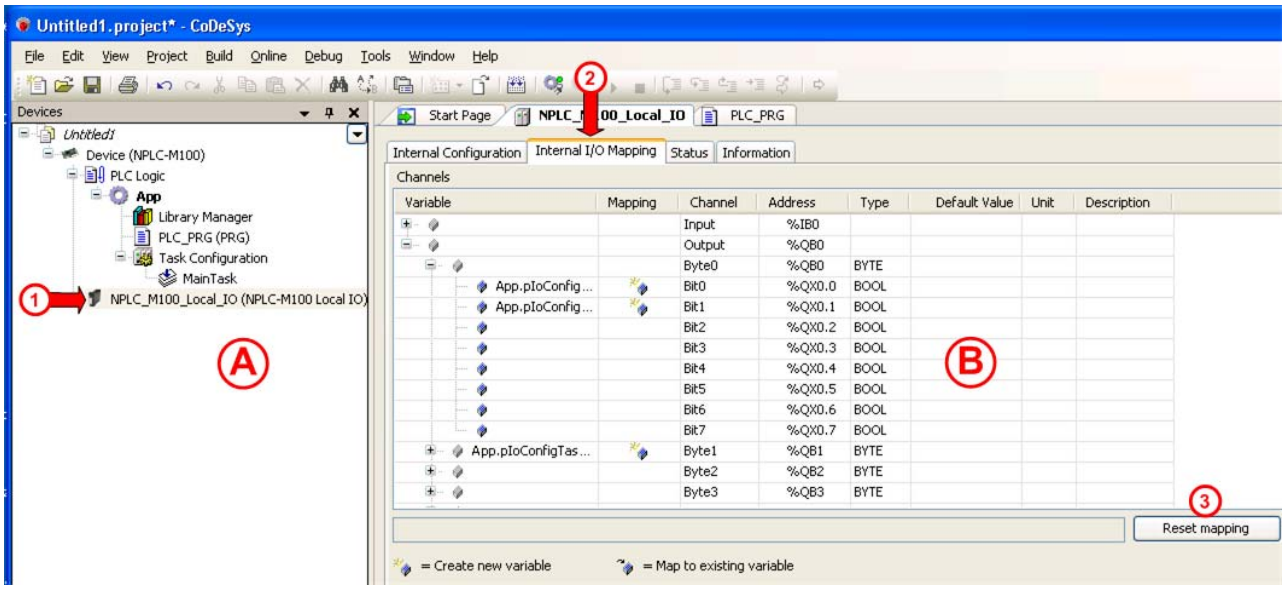


Figure 11: M100 Configure Local I/Os

- In the **Devices** tree (A) item **NPLC-M100 Local IO** (1) has been added.

© Hilscher, 2009-2014

- In the **Devices** tree double click **NPLC-M100 Local IO** ① to open signal configuration options in the main area Ⓑ.
- Select **Internal I/O Mapping** tab ②.

Now you can double click individual fields in the table, in order to configure signals for bits or words.

© Hilscher, 2009-2014

4.4 Create Bus Configuration

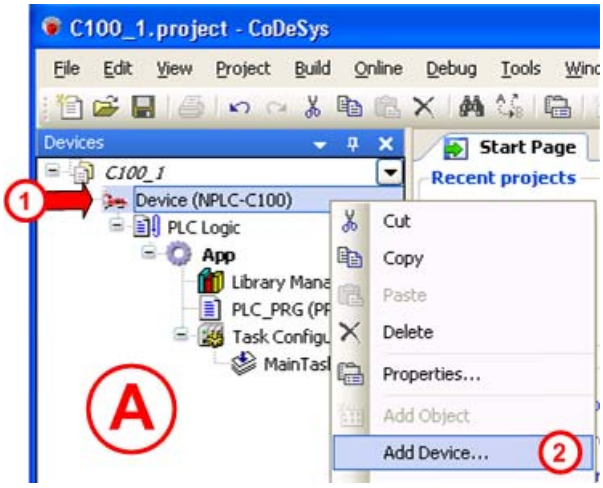
For your system, only one bus configuration is required. Thus, sections *Configure PROFIBUS Master*, *Add CANopen Master Configuration Window* and *Configure DeviceNet Scanner* are to be considered as alternatives.

Note: The CoDeSys online help provides instructions for bus configuration in its chapter **Editors > Device Editors > [fieldbus name]**.

4.4.1 Select Fieldbus System for the Controller

1. Select Fieldbus system for the controller.

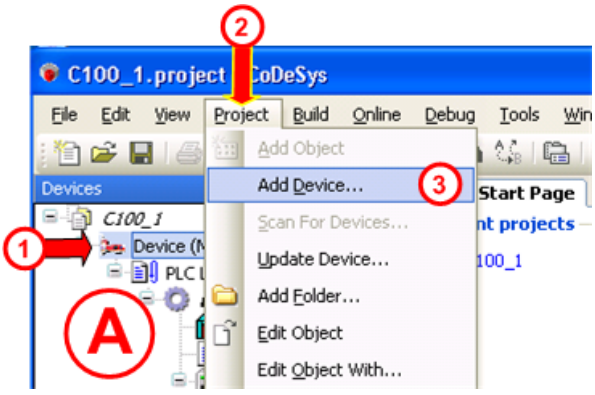
➢ In the **Devices** tree Ⓐ right click the device ① to open the context menu.



➢ In the context menu select **Add Device…** ②.

or:

➢ In the **Devices** tree Ⓐ select the device ①.



➢ In the **Project** menu ② select **Add Device…** ③.

➢ The following window opens:

© Hilscher, 2009-2014

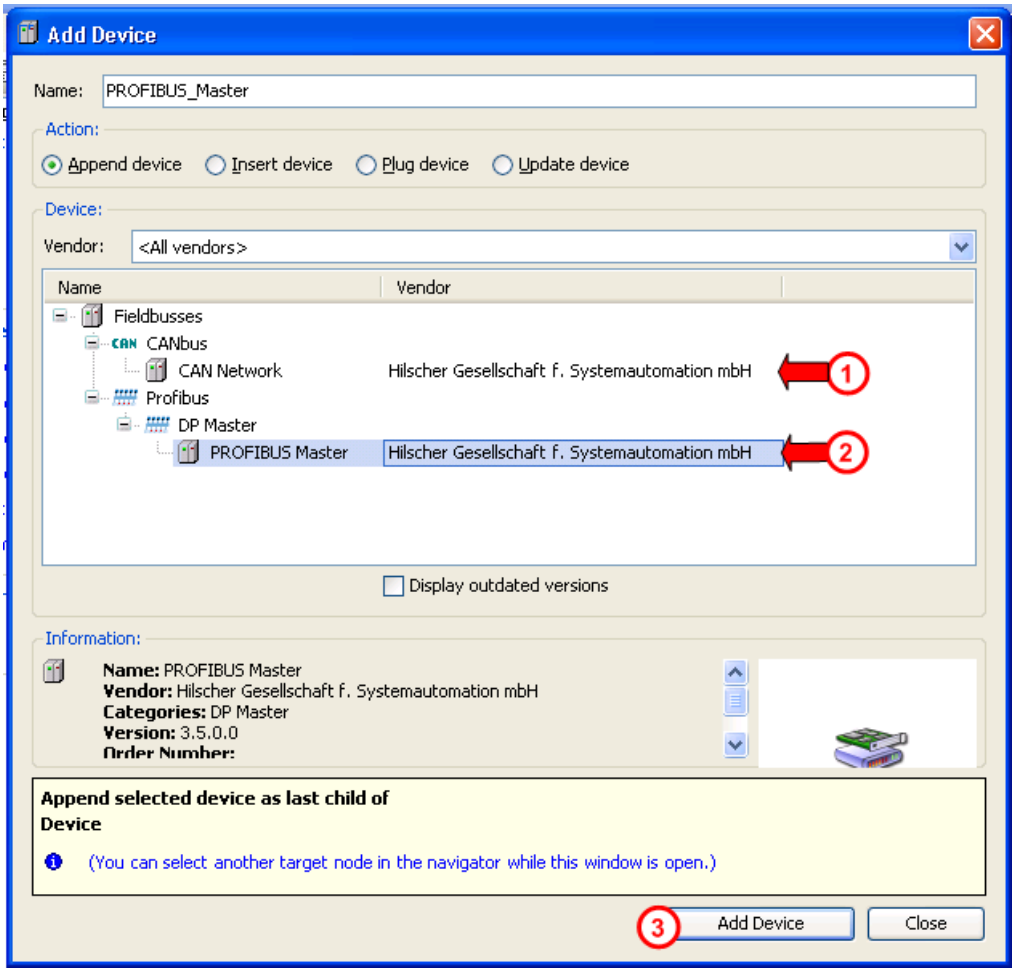


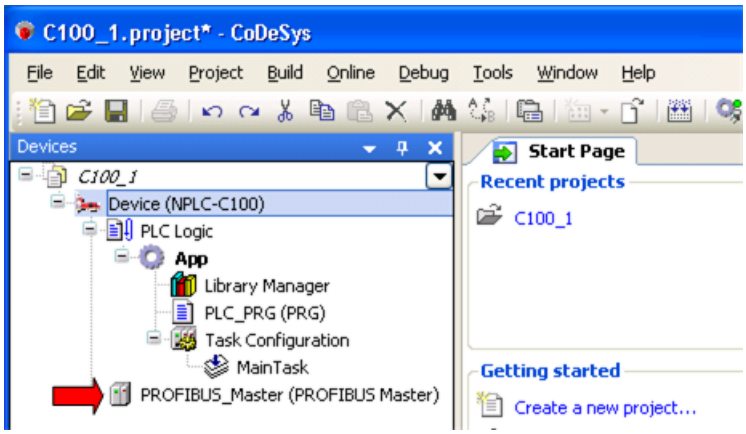
Figure 12: Select Fieldbus

2. Choose Fieldbus.

➢ For CANopen or DeviceNet select **CAN Network** ①.

For PROFIBUS select **PROFIBUS Master** ②.

➢ Click **Add Device** ③.



In the **Devices** tree an entry for the bus system has been added.

For CANopen or DeviceNet, two steps are needed.

© Hilscher, 2009-2014

4.4.2 Configure PROFIBUS Master

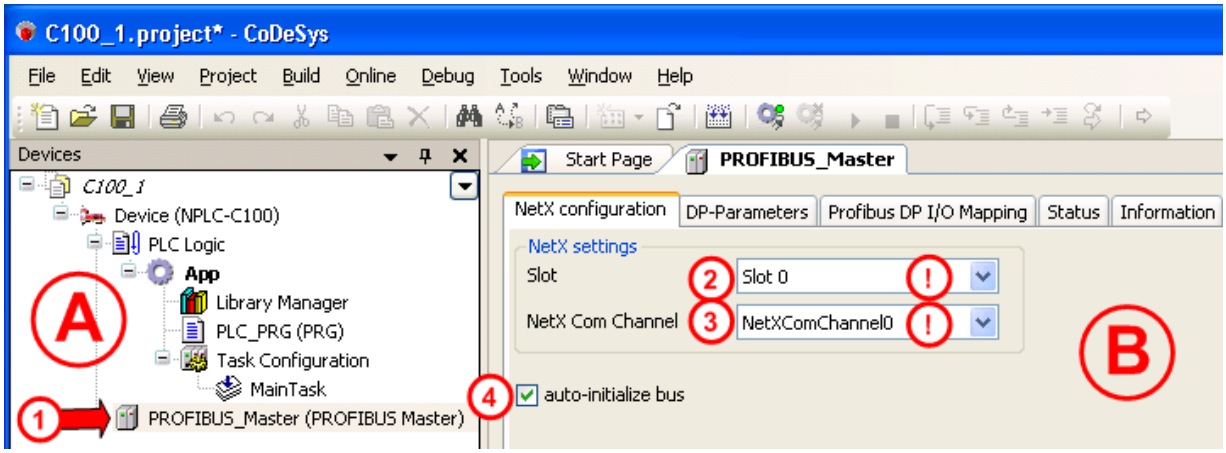


Figure 13: PROFIBUS Master NetX Configuration

- In the **Devices** tree Ⓐ double click **PROFIBUS Master** ① to open the configuration dialog for PROFIBUS parameters.

In the main area Ⓑ the first tab **NetX configuration** opens.

All standard parameters shown above must remain in their default settings:

- **Slot 0** ②,
- **NetXComChannel0** ③,
- **auto-initialize bus** ④ must remain checked.

© Hilscher, 2009-2014

In the **DP-Parameters** tab, you can set the following parameters:

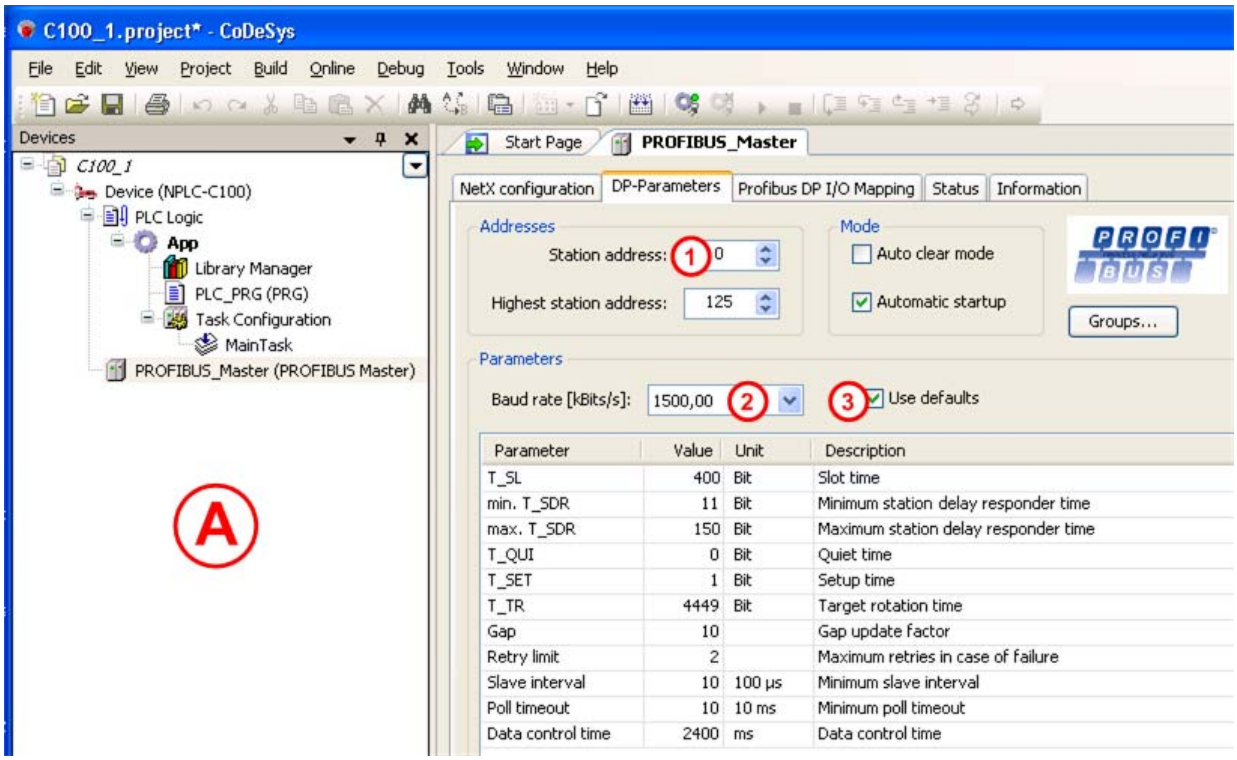


Figure 14: PROFIBUS Master DP Parameter Configuration

- Set the **Station address** ①.
- Set the **Baud rate** ②.
- All other parameters can only be set after the box next to **Use defaults** ③ has been unchecked.

© Hilscher, 2009-2014

4.4.3 Add CANopen Master Configuration Window

The configuration of a CANopen Master is done in several steps.

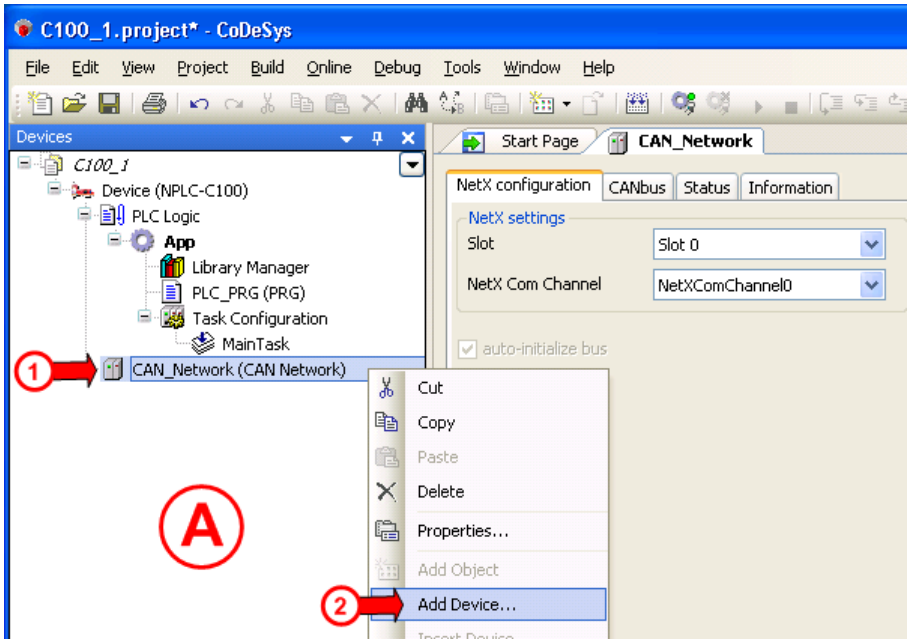


Figure 15: CANopen 2. Open Add Device Dialog

After the CAN bus has been added to the **Devices** tree according to the instructions given in section *Create Bus Configuration* on page 25,

- ➢ right click the **CAN_Network** ① in the **Devices** tree Ⓐ to open context menu.
- ➢ In the context menu select **Add Device…** ②.
- ➪ The **Add Device** dialog opens.

© Hilscher, 2009-2014

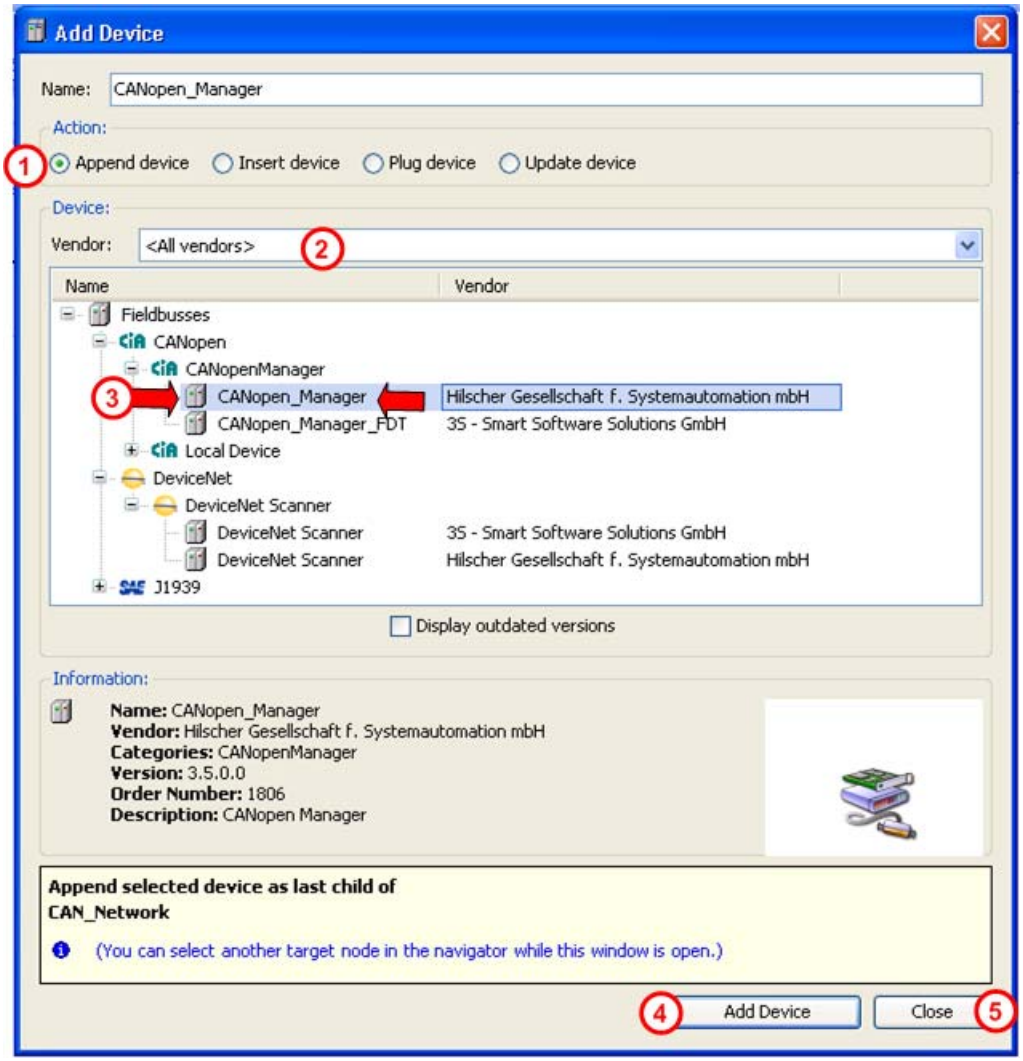


Figure 16: CANopen Manager

Set the following parameters:

- Select **Action** > **Append device** (1).
- Select **Device** > **<All vendors>** (2).
- Select **CANopen_Manager** (3).
- Click **Add Device** (4).
- Click **Close** (5).

The dialog window closes.

You have gathered all dialog windows for the configuration of the CANopen bus.

© Hilscher, 2009-2014

4.4.3.1 Configure CANopen Master

The Device tree has been expanded as follows:

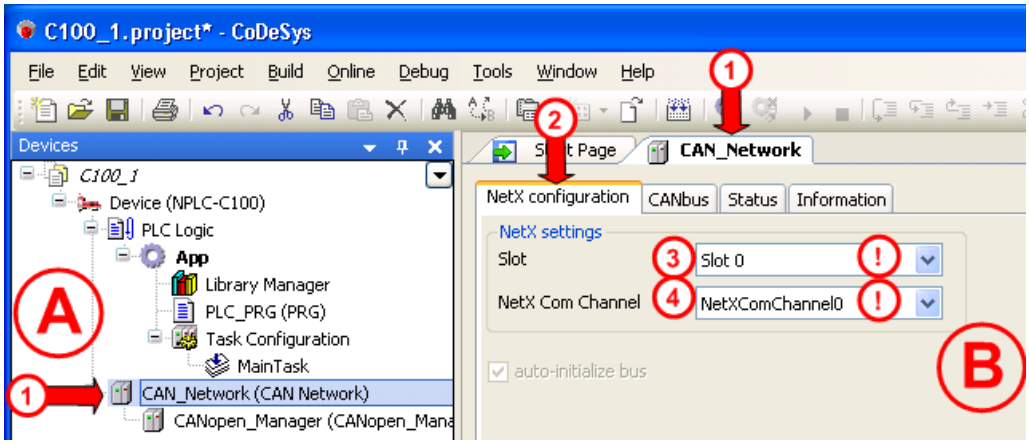


Figure 17: NetX_CANbus > NetX Configuration

Under **CAN_Network** (1) > **NetX Configuration** (2) set the following parameters:

- **Slot:** Always select **Slot 0** (3).
- **NetX Com Channel** (4): Always select **NetXComChanel0**.

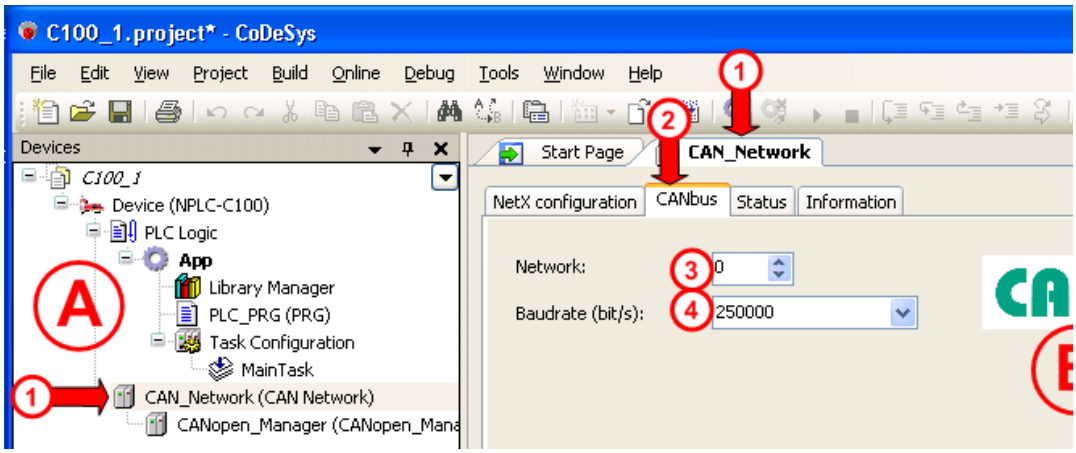


Figure 18: NetX_CANbus > CANbus

Under **CAN_Network** (1) > **CANbus** (2) set the following parameters:

- **Network** (3): Always select **0**.
- **Baudrate** (4): You can choose one of the following rates: 10000, 20000, 50000, 100000, 125000, 250000, 500000, 800000 and 1000000.

© Hilscher, 2009-2014

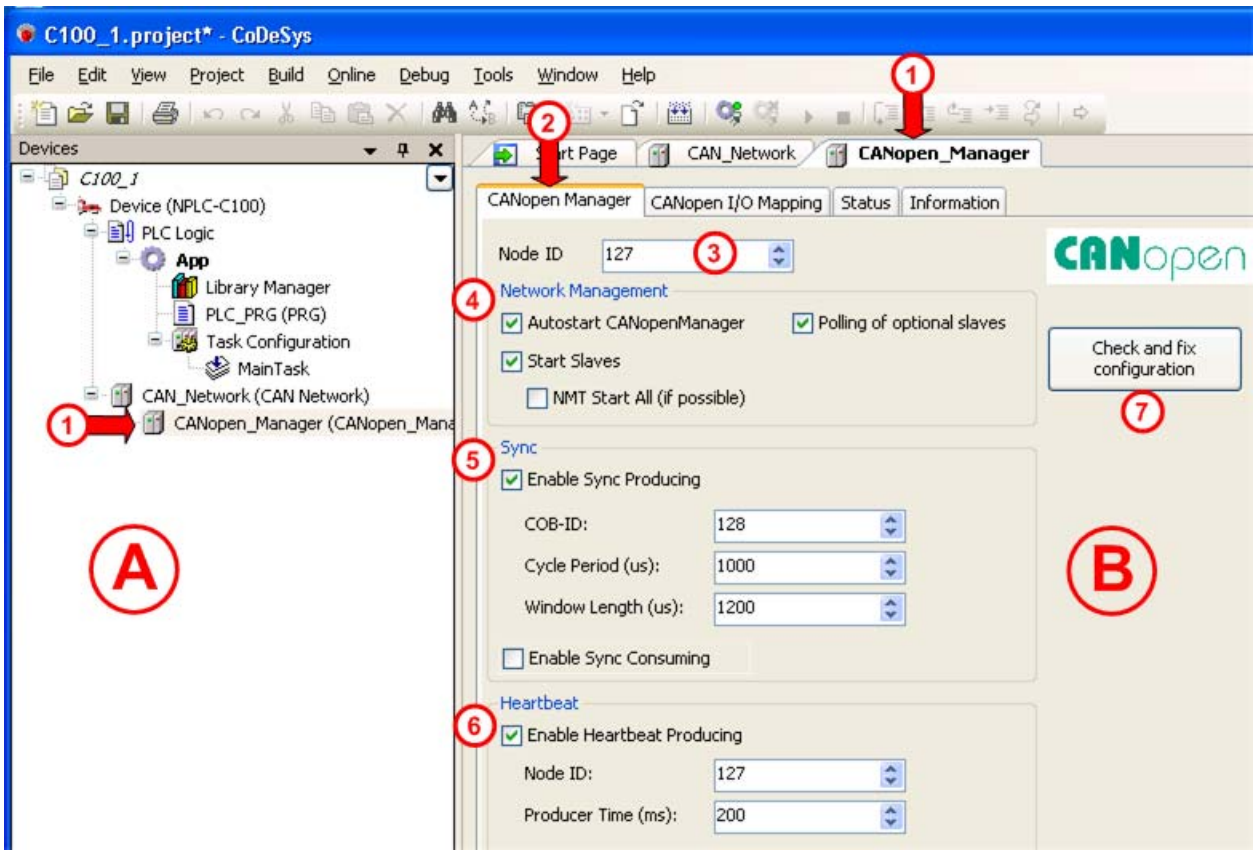


Figure 19: CANopen_Manager > CANopen_Manager

In the **Devices** tree Ⓐ double click **CANopen_Manager** ① to open the configuration dialog.

Under **CANopen_Manager** ① > **CANopen_Manager** ② set the following parameters:

- Node-ID ③
- Network Management ④
- Sync Behavior ⑤
- Heartbeat Producing ⑥

After having set the parameters, you can check the settings for plausibility by clicking **Check and fix configuration** ⑦.

© Hilscher, 2009-2014

4.4.4 Configure DeviceNet Scanner

The configuration of a DeviceNet Scanner is done in several steps.

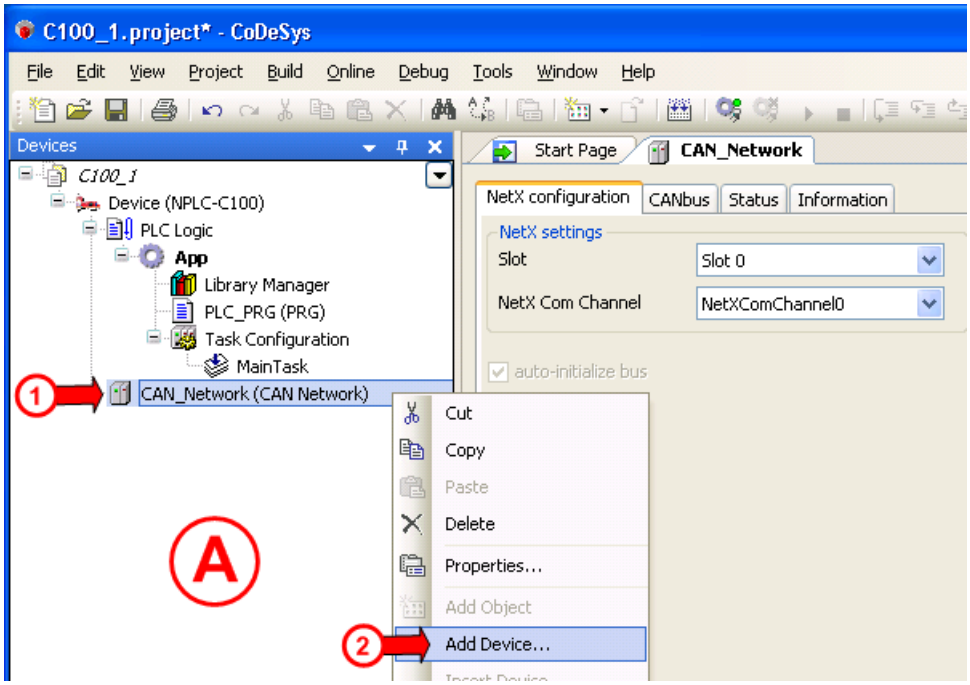


Figure 20: CANopen > Add Device

After the CAN bus has been added to the **Devices** tree according to the instructions given in section *Create Bus Configuration* on page 25,

- ➢ right click **CAN_Network** (1) in the **Devices** tree (A) to open the context menu.
- ➢ In the context menu select **Add Device…** (2).
- ➪ The **Add Device** dialog opens:

© Hilscher, 2009-2014

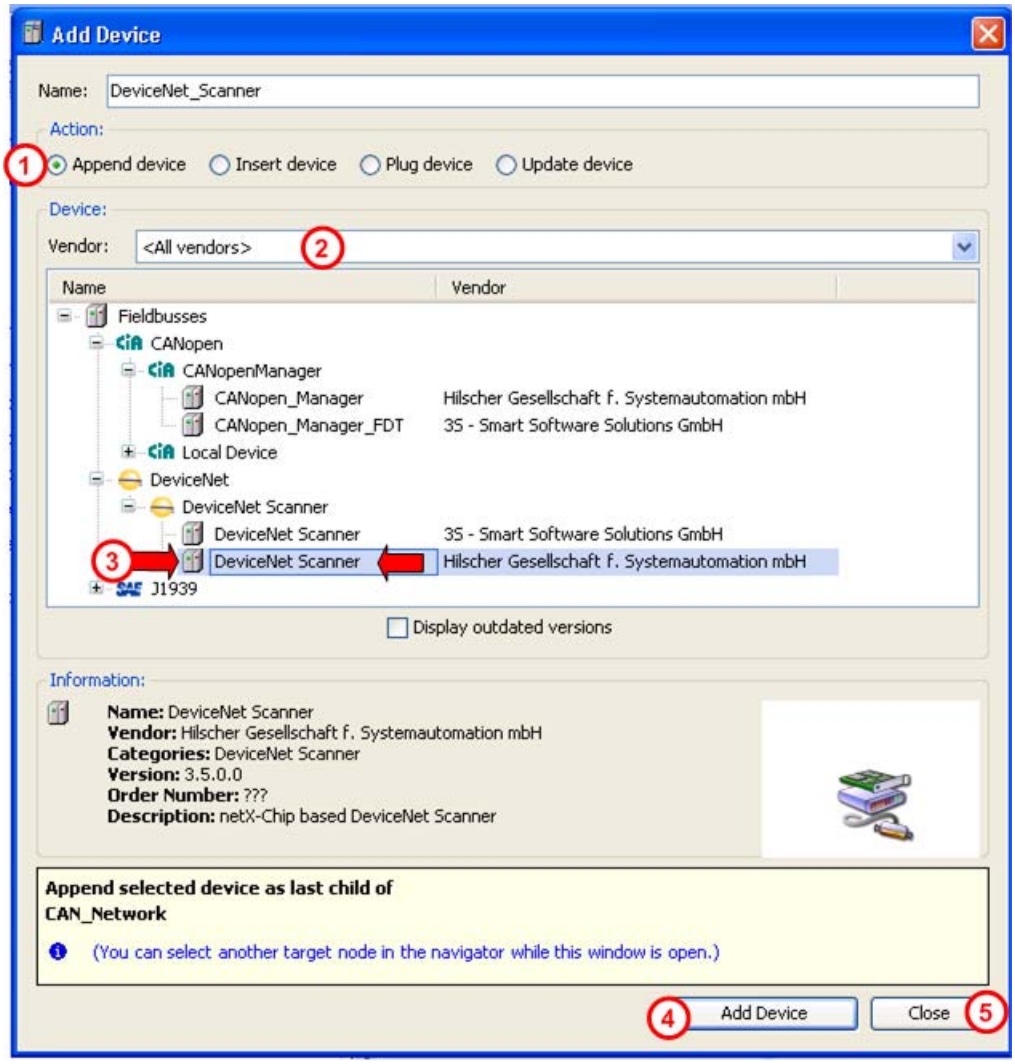


Figure 21: DeviceNet Scanner

Set the following parameters:

- Select **Action > Append device** (1).
- Select **Device** > **<All vendors>** (2).
- Select **DeviceNet Scanner** (3).
- Click **Add Device** (4).
- Click **Close** (5).

The dialog window closes.

You have gathered all dialog windows for the configuration of the CAN network.

© Hilscher, 2009-2014

The Device tree has been expanded as follows:

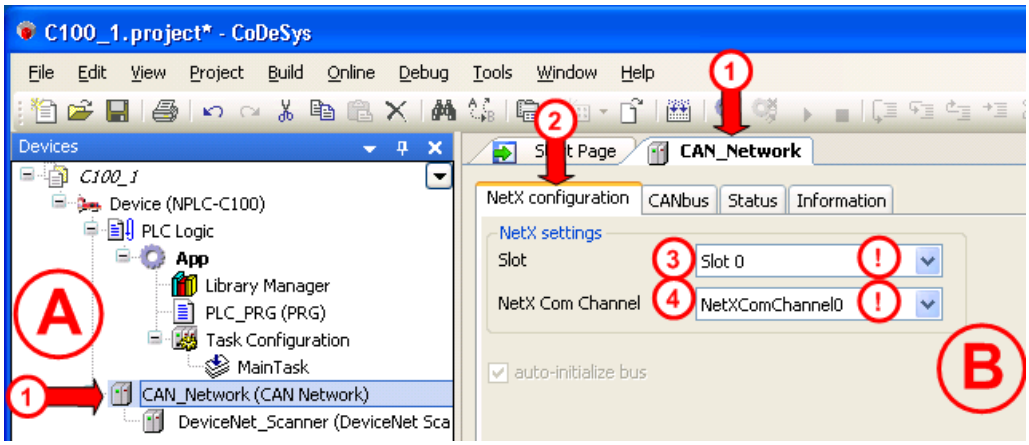


Figure 22: NetX_CANbus > NetX Configuration

Under **CAN_Network** ① > **NetX Configuration** ② set the following parameters:

- **Slot** ③: Always select **Slot 0**.
- **NetX Com Channel** ④: Always select **NetXComChanel0**.

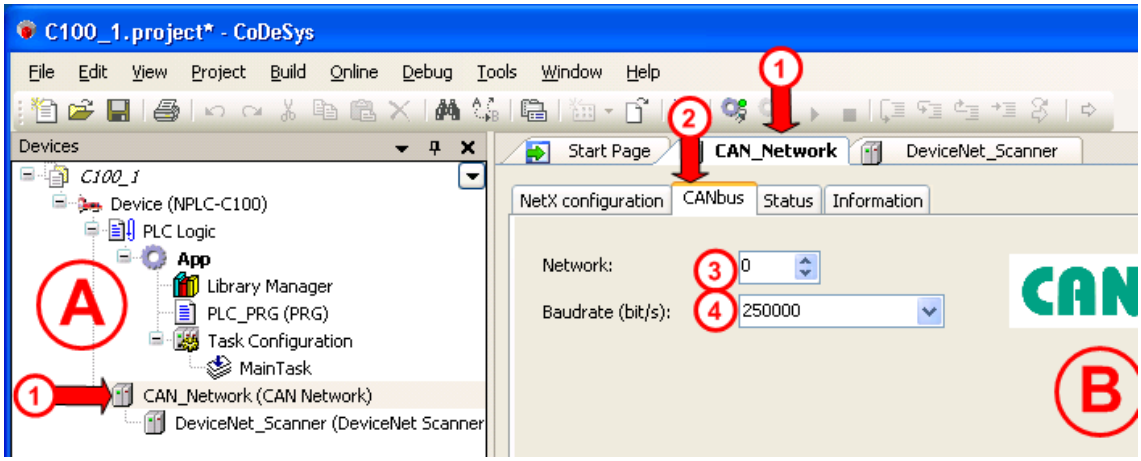


Figure 23: NetX_CANbus > CANbus

Under **CAN_Network** ① > **CANbus** ② set the following parameters:

- **Network** ③: Always select **0**.
- **Baudrate** ④: You can choose one of the following rates: 125000, 250000, 500000.

© Hilscher, 2009-2014

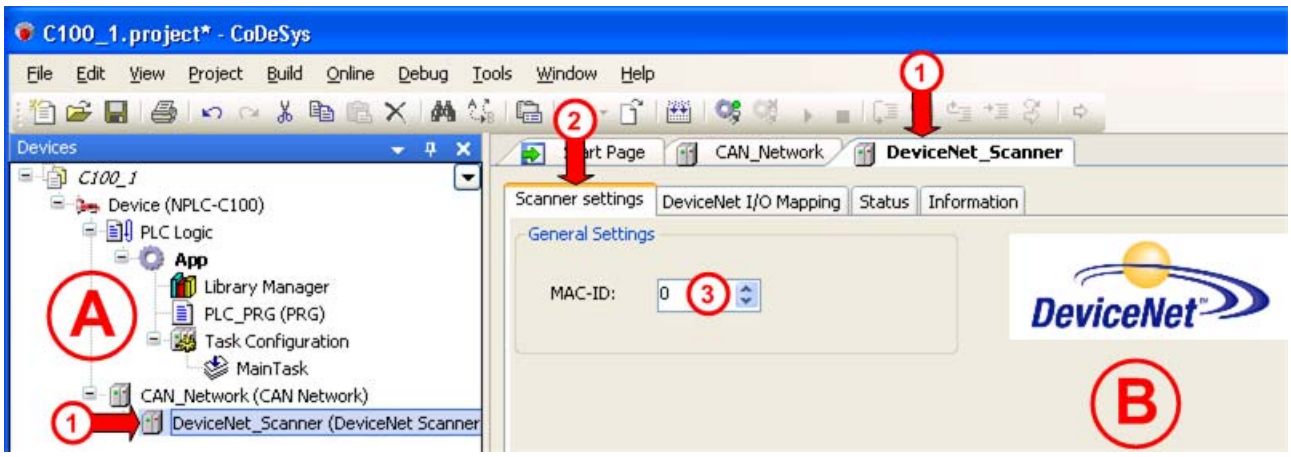


Figure 24: Set Fieldbus Address of Device

In the **Devices** tree Ⓐ double click **DeviceNet_Scanner** ① to open the configuration dialog.

Under **DeviceNet_Scanner** ① > **Scanner Settings** ② set the following parameter:

- **MAC-ID** ③: Set the fieldbus address of device.

© Hilscher, 2009-2014

4.5 Map Bus Structure

4.5.1 Import Slave Device Files

This section describes the import of slave devices based on device description files.

A PROFIBUS system with a slave is used for example here.

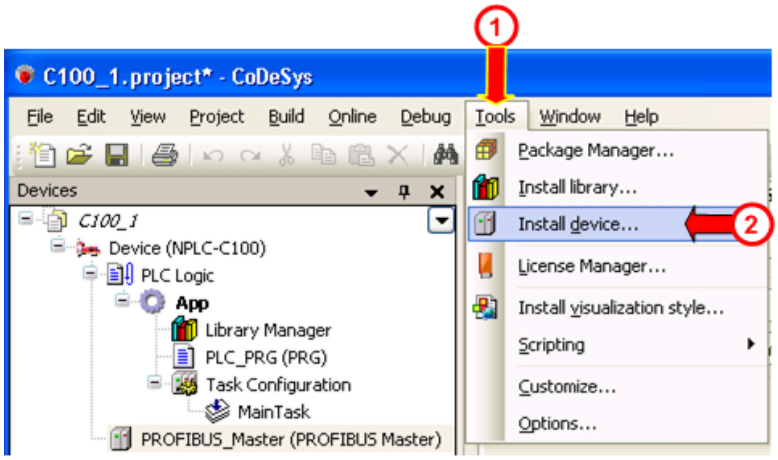


Figure 25: Import Device Description File

- In the **Tools** menu (1) select **Install Device** (2) to open file selection dialog.
- In the file selection dialog select the device description file.
- In the file selection dialog click **Open**.
- The device description file is imported.
- Now it is possible to add this slave device into the network configuration.

© Hilscher, 2009-2014

4.5.2 Add and configure Slave Device

The integration of the bus structure is exemplified by using PROFIBUS. The procedure is the same for other bus systems.

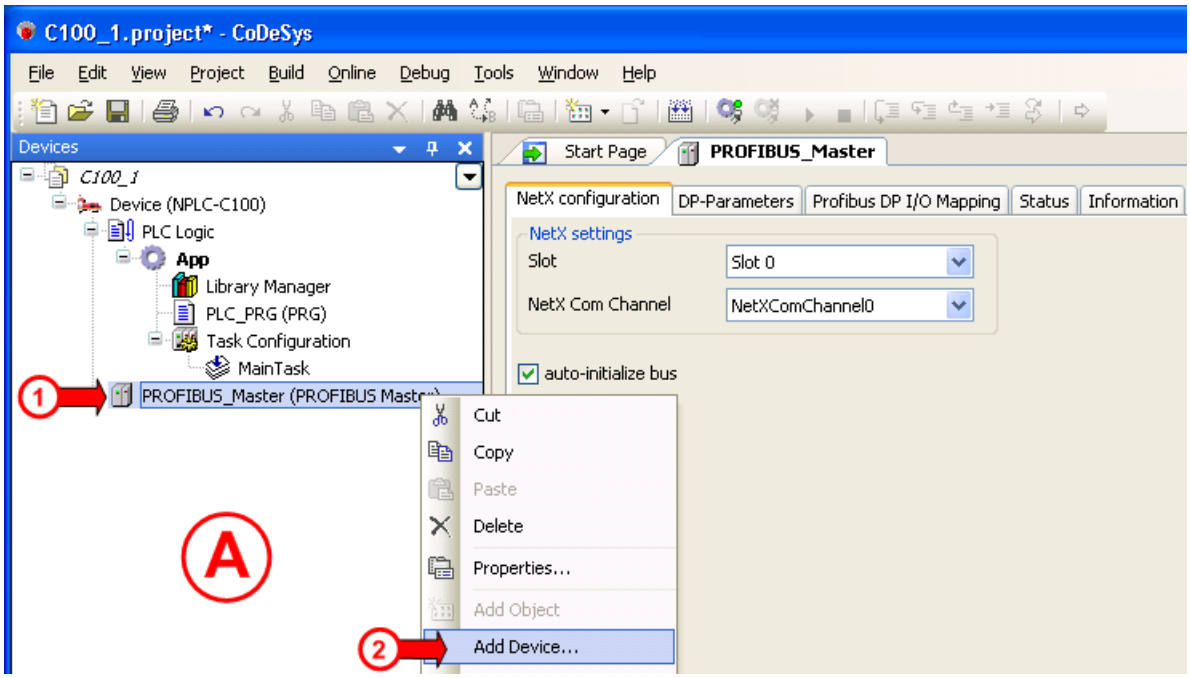


Figure 26: Add Slave Device

- In the **Devices** tree Ⓐ right click **PROFIBUS_Master** ①.
- Select **Add Device**.
- The **Add Device** window opens:

© Hilscher, 2009-2014

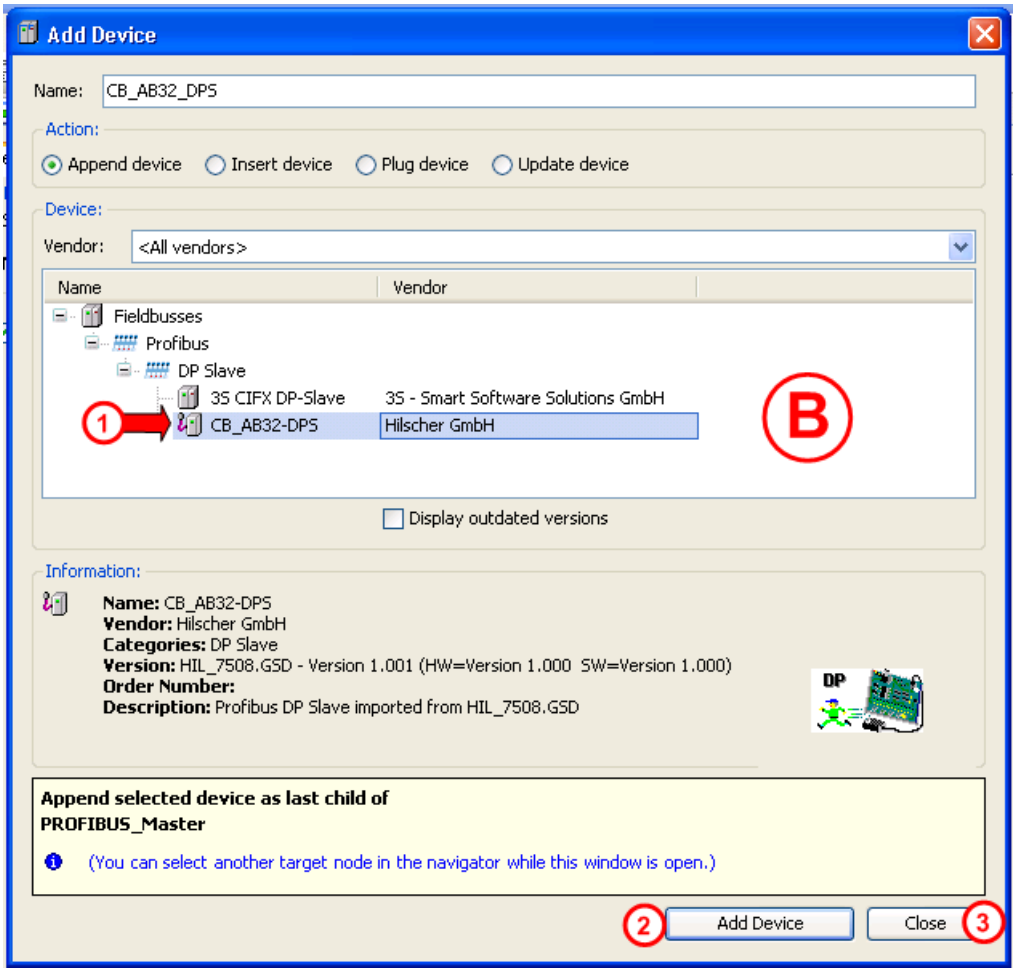


Figure 27: Select Slave Device

The Device area (B) (see figure above) lists all devices available for the fieldbus system.

- Select your slave device (1).
- Click **Add Device** (2).
- Click **Close** (3).

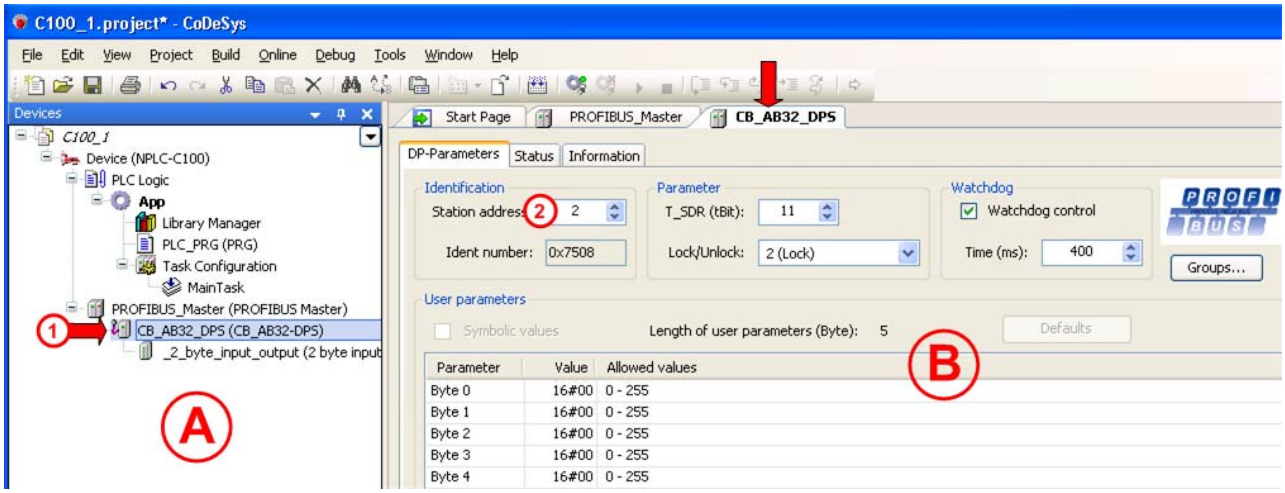


Figure 28: Added Slave Device

© Hilscher, 2009-2014

In the **Devices** tree Ⓐ the entry for the slave device ① has been added.

➢ Double click slave device ①.

⇛ A configuration dialog for the slave opens in the main area Ⓑ.

➢ Set here all necessary parameters.

➢ Set the **Station address** ② for the slave device.

Note: Here you only set parameters which are necessary for the master to be able to communicate with the slave. The slave device itself must be configured by using the adequate slave device configuration tool.

Note: Right click on a device in the **Devices** tree Ⓐ to open via the context menu the **Properties** dialog of the device. Here you can choose a different name for the device to be displayed in the **Device** tree.

4.5.3 Identify I/O Addresses of the Slave

To find out the input/output addresses of the slave, which are to be used in the PLC program, proceed as follows:

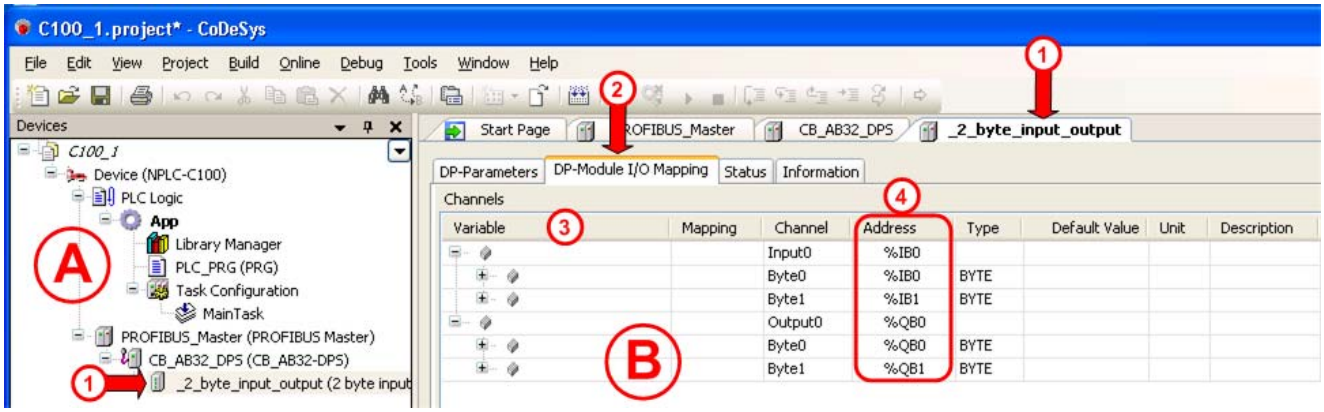


Figure 29: Ascertain I/O Addresses of Slave

➢ In the **Device** tree Ⓐ double click the I/O entry ① of the slave.

⇛ The dialog view of the slave I/O opens in the main area Ⓑ (see figure above).

➢ Click **DP-Module I/O Mapping** tab ②.

➢ In the **Variable** column ③ click on the plus symbols to expand the list where necessary.

⇛ In the **Address** column ④ you can see the values which can be used in the PLC program to address the I/Os.

Note: The addresses ④ are %IB0, %IB1, %QB0 and %QB1, when no Host I/O interface is used. The addresses ④ will be %IB512, %IB513, %QB512 and %QB513, when Host I/O interface is used in front of the master device and its slave devices.

© Hilscher, 2009-2014

4.6 Create PLC Program

1. Open Editor for function module PLC_PRG

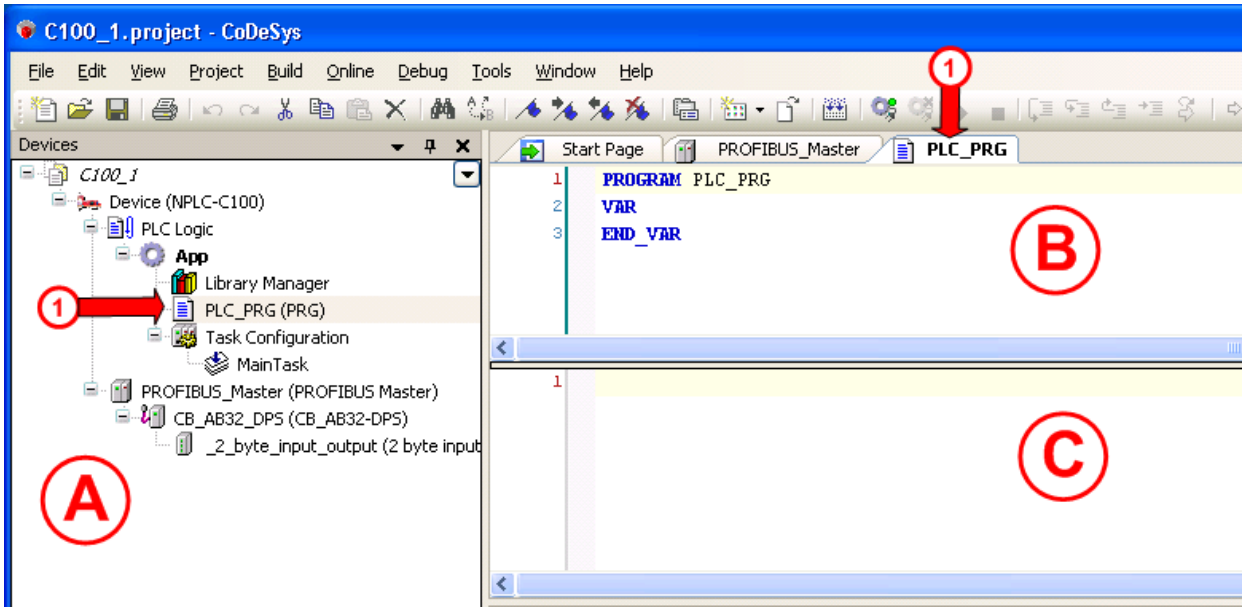


Figure 30: Slave Device Added

➢ In the Devices tree Ⓐ double click **PLC_PRG (PRG)** ①.

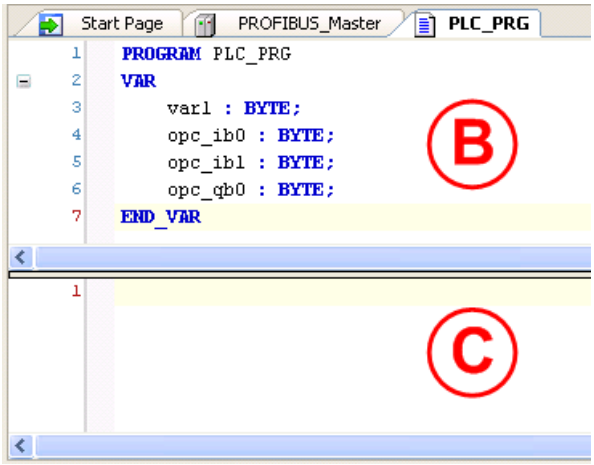
Note: You can use the program editor only if you are not logged into the PLC component.

➢ The editor window appears. The following figure shows the Editor without any variables Ⓑ and without PLC program Ⓒ.

2. Create a variable.

➢ Create following variables:

```
var1 : BYTE;
opc_ib0 : BYTE;
opc_ib1 : BYTE;
opc_qb0 : BYTE;
```

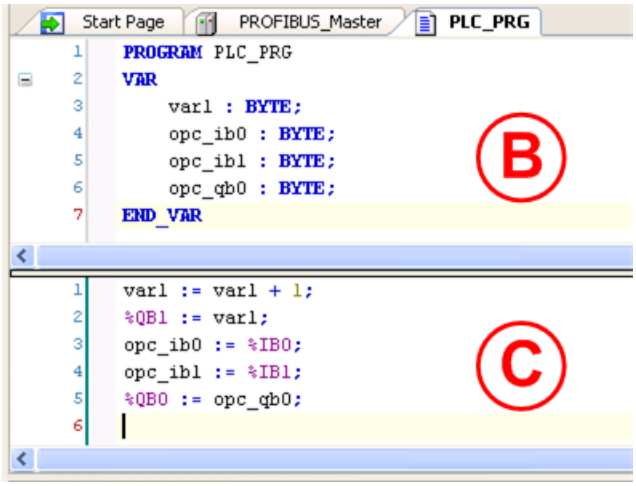


© Hilscher, 2009-2014

3. Edit PLC program.

➢ Fill in the following PLC program:

```
var1 := var1 + 1;
%QB1 := var1;
opc_ib0 := %IB0;
opc_ib1 := %IB1;
%QB0 := opc_qb0;
```



A short explanation of the programming example:

`var1 := var1 + 1;`

Variable var1 is incremented each PLC cycle.

`%QB1 := var1;`

The value of variable var1 is written to the output byte %QB1.

`opc_ib0 := %IB0;` and `opc_ib1 := %IB1;`

Input bytes %IB0 and %IB1 are copied to the variables opc_ib0 and opc_ib1. This is the preparation for the OPC server to be able to access the input data.

`%QB0 := opcqb0;`

The value of variable opc_qb0 is copied to the output byte %QB0. This is the preparation for the OPC server to be able to access the output data.

Remark: Output byte %QB1 is set by the PLC program and therefore can't be accessed by the OPC server.

Note: The addresses %IB0 and %QB0 (area Ⓒ) starts from 0, because no Host I/O interface is used. When Host I/O interface is used in front of the master device and its slave devices, then the addresses will start from 512 and thus are %IB512 and %QB512 (and higher).

4. Save the project.

➢ Save project using menu entry **File > Save Project**.

➢ The project is saved.

© Hilscher, 2009-2014

To create retain variables:

1. Create retain variables

➢ Create the section
VAR RETAIN

END_VAR

➢ Create a variable inside this section
VAR RETAIN
uiRemCount : UINT;
END_VAR

```
1  PROGRAM PLC_PRG
2  VAR
3      var1 : BYTE;
4      opc_ib0 : BYTE;
5      opc_ib1 : BYTE;
6      opc_qb0 : BYTE;
7  END_VAR
8  VAR RETAIN
9      uiRemCount : UINT;
10 END_VAR
```

Note: Persistent variables can be created via the context menu of the **APP** entry in the **Devices** tree.

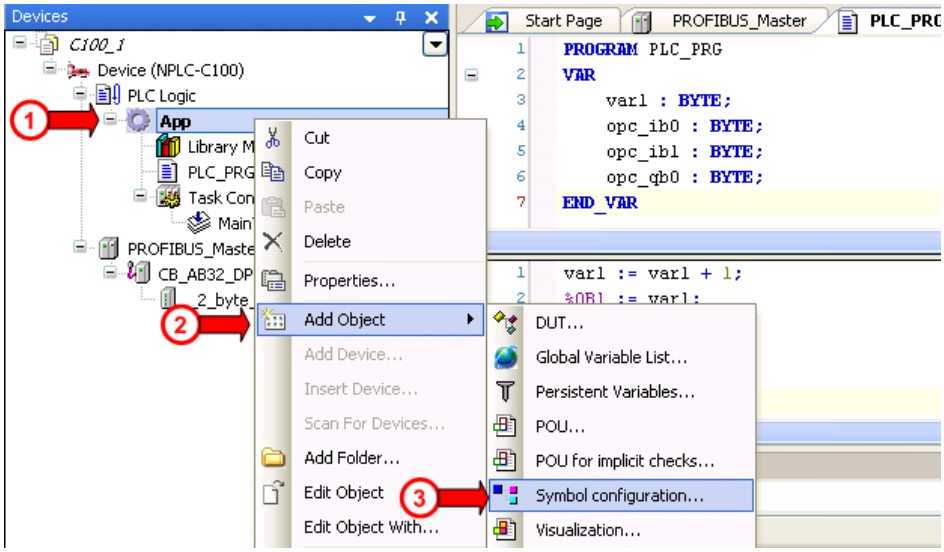
For more information on retain and persistent variables, please refer to the CoDeSys online help.

© Hilscher, 2009-2014

4.6.1 Create Symbol Configuration

Symbol configuration is used to create symbols with certain access rights, which in turn can be used to access project (application) variables from the outside, e. g. from an OPC client.

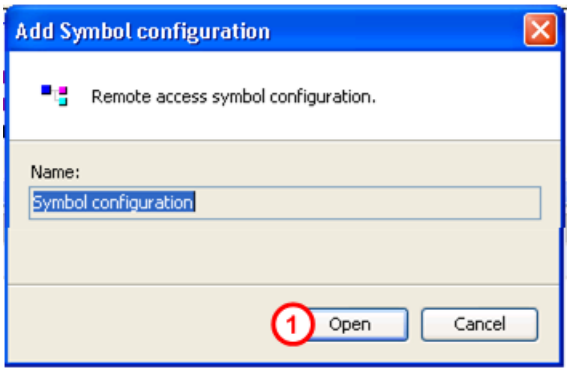
1. Add object.

➢ 

In the **Devices** tree, right click **App** entry to open context menu, then select **Add Object**, then select **Symbol configuration**.

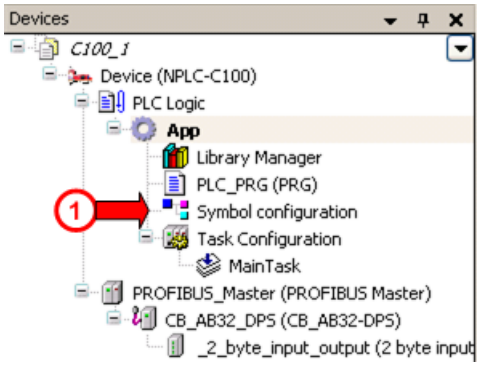
2. Add object symbol configuration.

➢ The following dialog window opens:



➢ Click **Open** (1).

➢ In the **Devices** tree the **Symbol Configuration** entry appears.



➢ In the editor window the **Symbols** configuration tab appears.

© Hilscher, 2009-2014

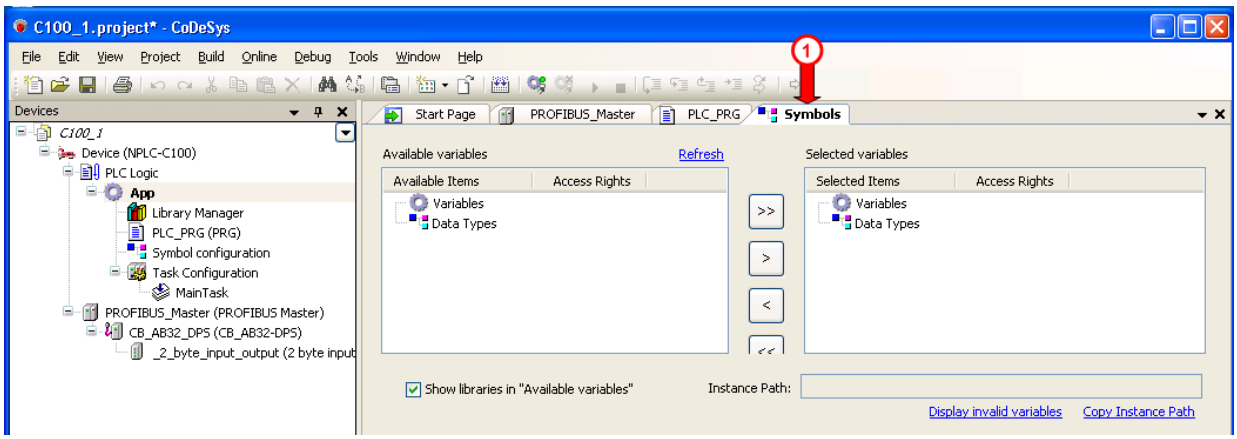
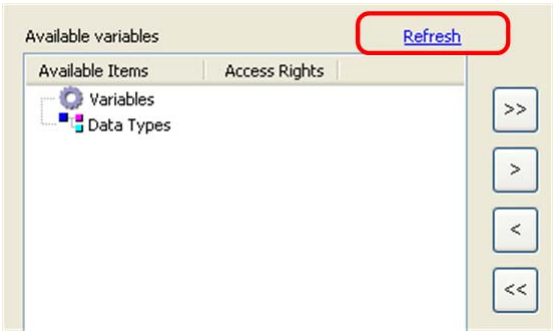


Figure 31: Editor Symbols

3. Refresh available variables.

➢ In the **Available Variables** area click **Refresh**.



➢ The **Available Variables** are updated.

4. Show available variables.

   In the **Available Variables** area Ⓑ the variables are displayed.

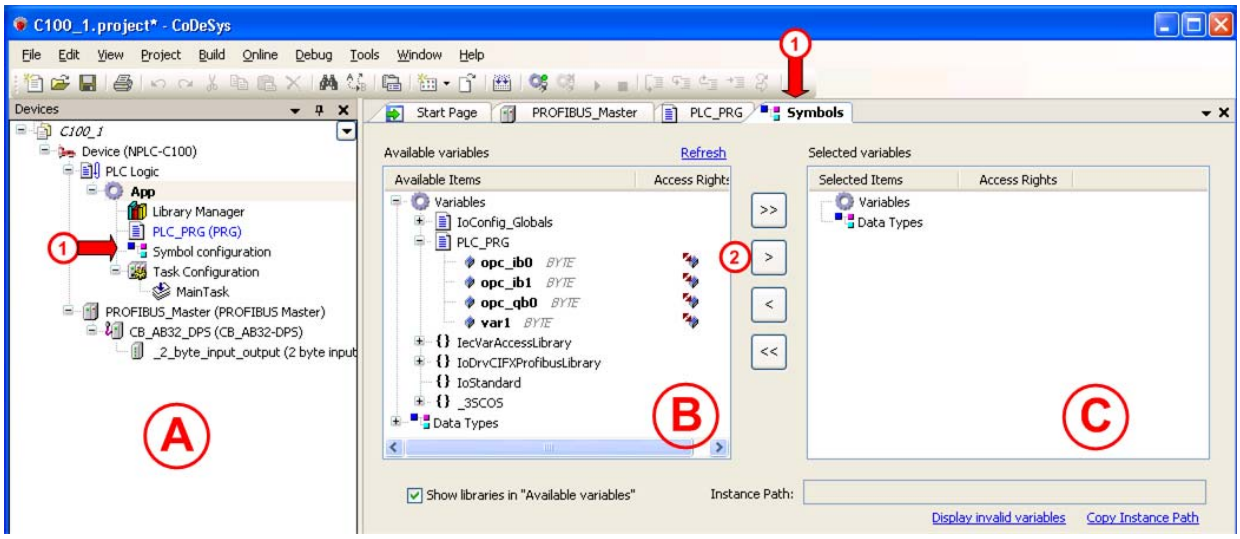


Figure 32: Editor Symbols (2)

5. Add symbols to selected variables.

➢ In the **Available Variables** tree Ⓑ select **opc_ib0** variable.

➢ Add this variable to the **Selected Variables** area Ⓒ by clicking "**>**" ②.

➢ Add in the same manner also variables **opc_ib1** and **opc_qb0** to the

© Hilscher, 2009-2014

Selected Variables area.

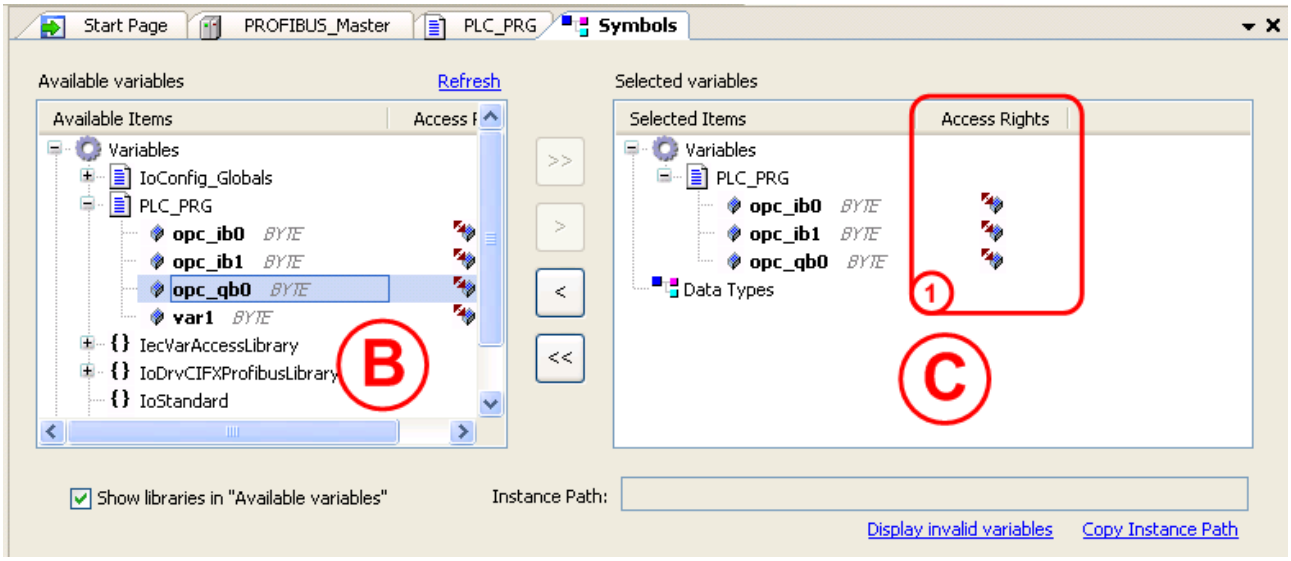


Figure 33: Editor Symbols Access Rights

In the **Selected Variables** area Ⓒ you can set Read/Write directions by clicking the symbols in the **Access Rights** column.

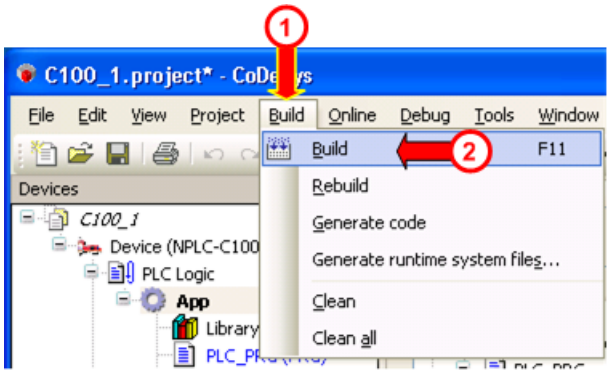
© Hilscher, 2009-2014

4.6.2 Build Project

Before loading the program to the netPLC component, the program has to be built.

1. Build Project.

➢ In the menu, choose **Build > Build [F11]** in order to build the project.



↬ The project is being built.

2. Check the message window.

➢ Check, if the message
Compile complete -- 0 errors, 0 warnings
appears in the message window:

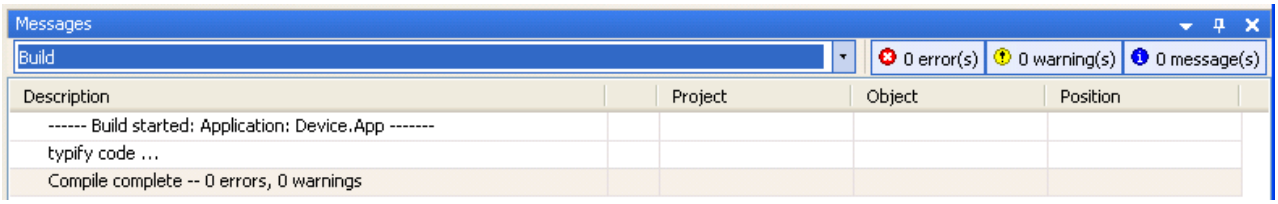


Figure 34: Messages After Building Process

↬ The project has successfully been built.

3. Save project.

➢ In the menu select **File > Save Project** to save the project.

↬ The project is saved.

© Hilscher, 2009-2014

4.7 Connect to the PLC

4.7.1 Define Gateway

1. Open the **Communication Settings** tab.

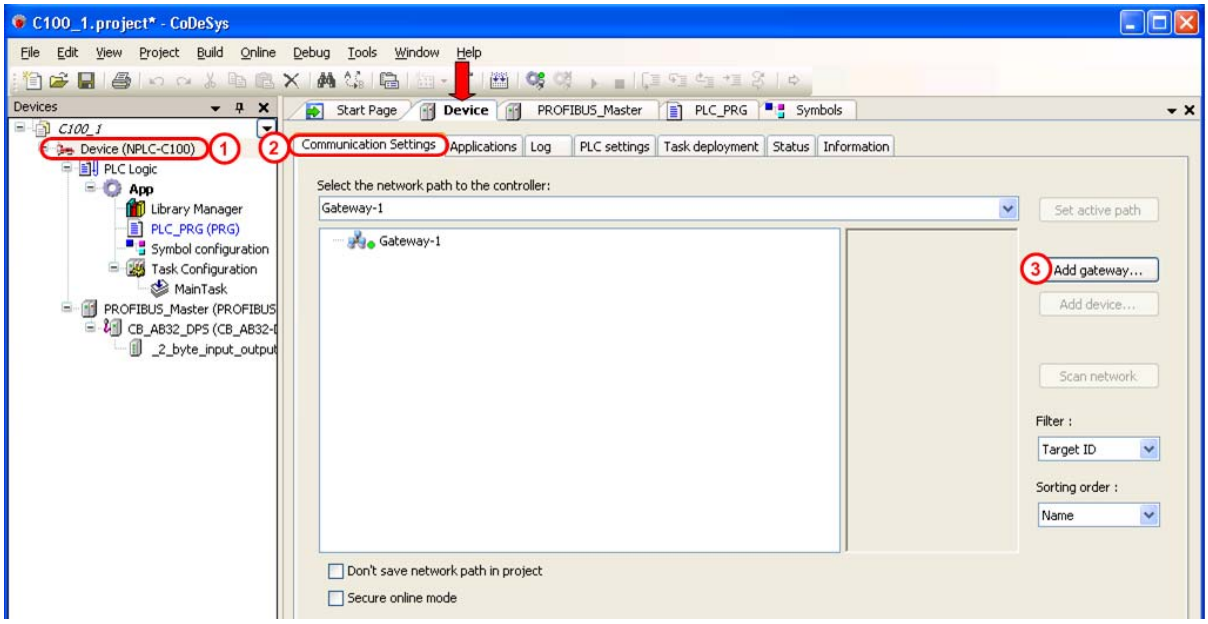


Figure 35: Communication Settings

➢ In the **Devices** tree double click **Device (NPLC-C100)** ① to open the **Communication Settings** tab ②, which is a sub tab of the **Device** tab.

➢ The **Device** tab with its **Communication Settings** sub tab is displayed.

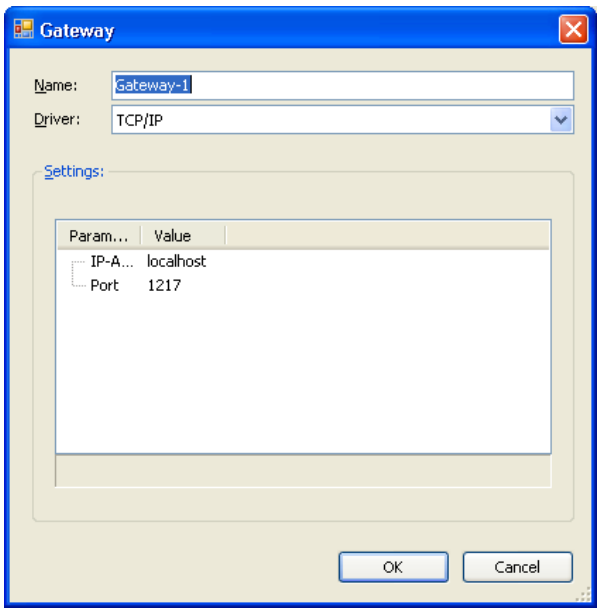
If no gateway is displayed in the tab, add a gateway as described in the following steps 2 and 3. If a gateway is displayed, skip steps 2 and 3 and proceed to the next section.

2. Add Gateway.

➢ Click **Add Gateway** ③.

➢ The **Gateway** window appears.

© Hilscher, 2009-2014



3. Accept Gateway.

➢ Click **OK** to take over **Gateway-1** settings.

➢ The window **Gateway** is closed.

4.7.2 Scan Network and Set Active Path

Note: Make sure, that the netPLC CoDeSys Server runs, when the NPLC-C100 card communicates via parallel PC system bus.

Note: If you want to connect to the netPLC component via USB, check if USB cable is connected.

1. Scan the network.

➢ Click **Scan Network** to search for PLC devices.

➢ Devices are searched via the USB connection, via the parallel system bus of the slot PLC, via the netPLC CoDeSys server as well as via Ethernet.

➢ Found devices (also non-PLC devices) are displayed below **Gateway-1**.

© Hilscher, 2009-2014

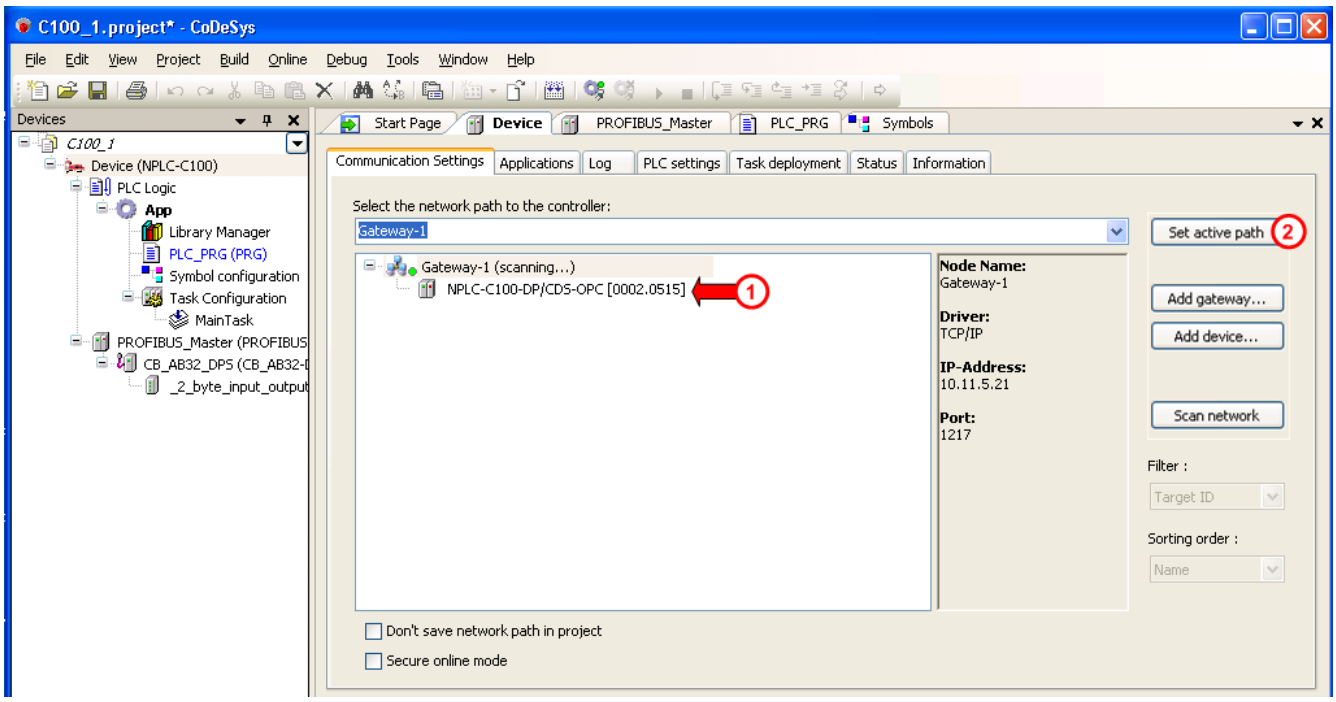


Figure 36: Set Active Path

2. Set active path.

➢ Select **NPLC-C100-DP** device (1) below Gateway-1.

➢ Click **Set active path** (2) to establish a connection to the device.

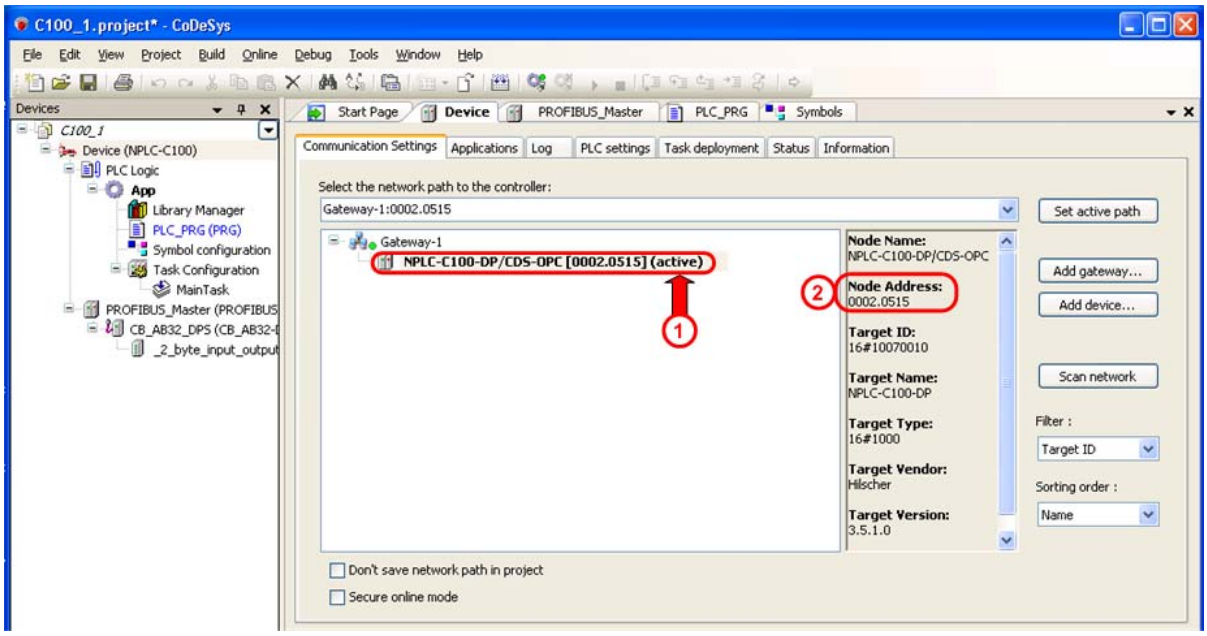


Figure 37: Path Set Active

➢ Next to the **NPLC-C100** device entry a message indicating **active** state appears (1).

Note: You need the **Node Address** (see position (2) in the figure above) to configure the OPC-Server.

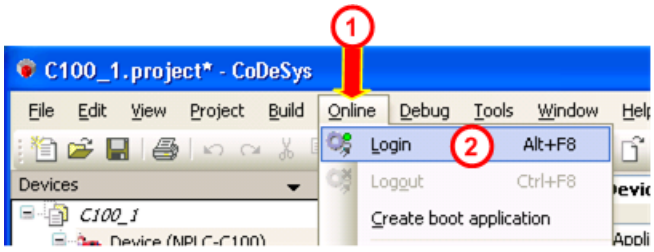
© Hilscher, 2009-2014

4.8 Download PLC Program and Bus Configuration

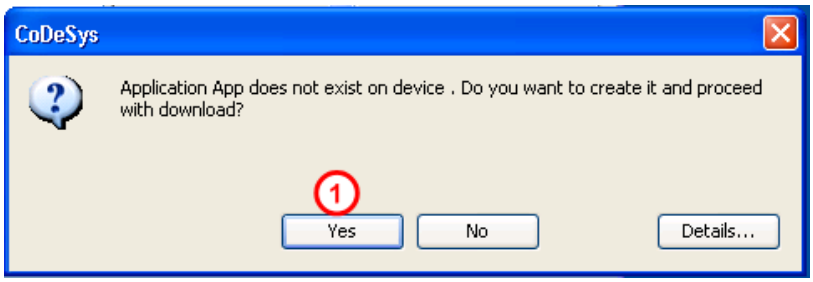
4.8.1 Login and Download

1. Login to the PLC.

➢ In the menu, choose **Online > Login**



The software verifies, if the project in the PLC component is the same as the project currently opened on the PC. If no PLC program is in the PLC component yet, then the message “Application App does not exist on device. Do you want to create it and proceed with download?“ is displayed. If the projects are different, the message “Unknown version of application App on target. Do you want to perform a download and replace the application?” is displayed.



Important: A download of the PLC program only by login does not store the PLC program in the PLC component zero voltage secure. If the netPLC component is disconnected from power supply, the downloaded PLC program and bus configuration are deleted. Thus it may happen, that an old program, which had been loaded to the remanent FLASH memory of the target component earlier before, is started again.

It is mandatory to create a boot application to store the PLC program and the bus configuration permanently (zero voltage secure) in the slot PLC. This is described in the *Create Boot Application* section on page 53.

2. Start download.

➢ Answer the question “Application App does not exist on device. Do you want to create it and proceed with download?“ with **Yes** (1).

The application is downloaded to the PLC.

With the start of the download, the PLC goes into stop mode.

The footer shows the progress of the download procedure: **Sending download info: Downloading …** and shows the size of the already downloaded program and the complete size of the program.

© Hilscher, 2009-2014

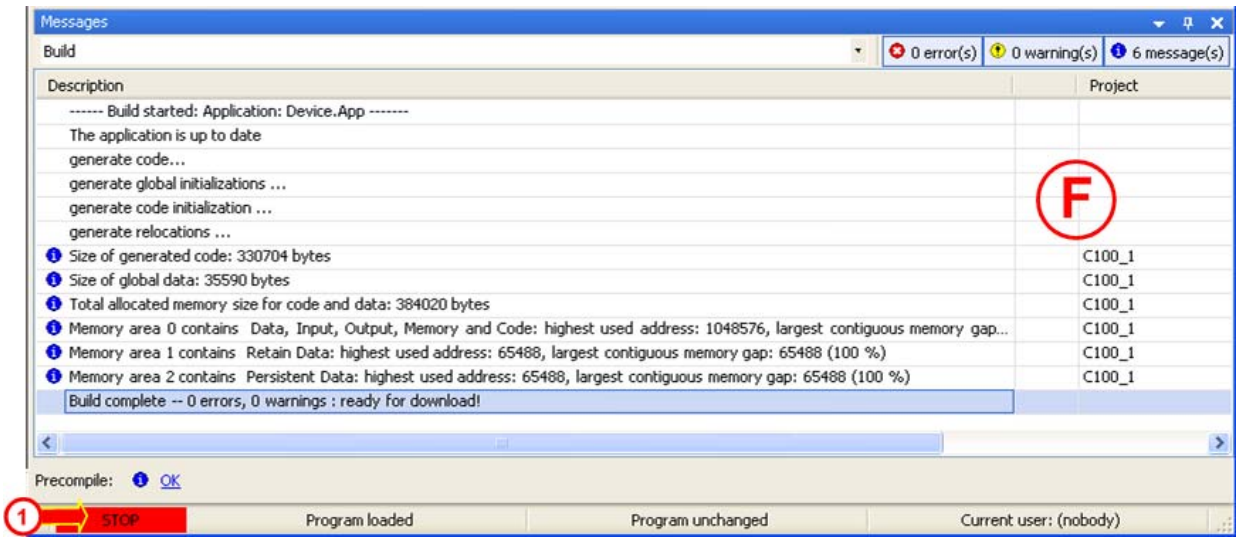

Messages
Build
0 error(s)
0 warning(s)
6 message(s)
Description
Project
------ Build started: Application: Device.App -------
The application is up to date
generate code...
generate global initializations ...
generate code initialization ...
generate relocations ...
Size of generated code: 330704 bytes
Size of global data: 35590 bytes
Total allocated memory size for code and data: 384020 bytes
Memory area 0 contains Data, Input, Output, Memory and Code: highest used address: 1048576, largest contiguous memory gap...
Memory area 1 contains Retain Data: highest used address: 65488, largest contiguous memory gap: 65488 (100 %)
Memory area 2 contains Persistent Data: highest used address: 65488, largest contiguous memory gap: 65488 (100 %)
C100_1
Build complete -- 0 errors, 0 warnings : ready for download!
F
Precompile:
OK
1
STOP
Program loaded
Program unchanged
Current user: (nobody)

Figure 38: Messages Building Process

➪ After download is complete, the footer shows the state of the loaded program and the PLC.

Important: The PLC remains in **Stop** mode. This is indicated in the footer of the **Messages** area (see position ① in the figure above). This is also indicated in the **Devices** tree behind the **App** entry.

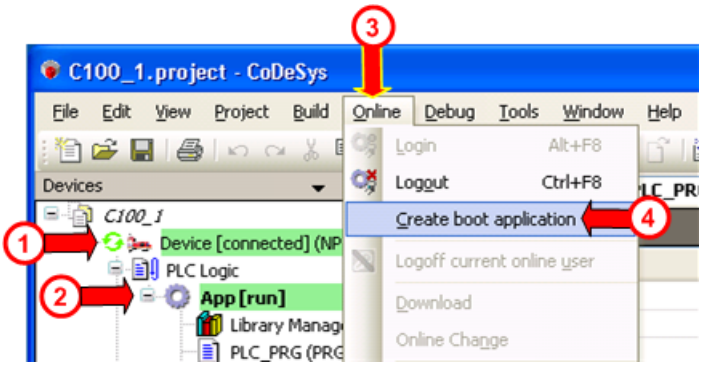
© Hilscher, 2009-2014

4.8.2 Create Boot Application

Important: It is mandatory to create a boot application to store the PLC program and the bus configuration permanently (zero voltage secure) in the netPLC component.

Note: In order to create a boot application on the netPLC component, you have to be connected online (logged in) to the component. If this is not the case, you will be asked to specify a location on the PC system, where you want to store the boot application.

Create boot application.



➢ Check, if the programming system is logged in. This is indicated by the green background color of the entries ① and ②.

For this, it is irrelevant, whether the application (the PLC program) is in **[Run]** or **[Stop]** mode.

➢ In the menu, choose **Online** ③ **> Create boot application** ④.

↬ The boot application is created and downloaded to the PLC.

4.8.3 Remove Boot Application

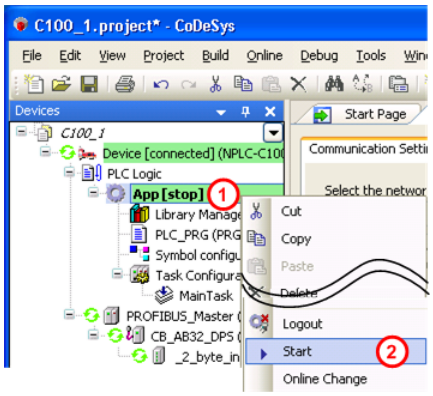
Remove boot application.

➢ If an existing boot application should be removed, select **Online > Reset > Origin**.

↬ The netPLC component is reset and the existing boot application is removed.

© Hilscher, 2009-2014

4.8.4 Start PLC Program

➢ 

In the **Devices** tree right click **App** (1) to open the context menu.

In the context menu click **Start** (2) to start the PLC program.

➢ **Important:** Make sure, that the hardware switch of the netPLC component (if the component is endowed with this switch) is in RUN position. If it is in STOP position, switch it to RUN.

➢ The PLC is in state RUN.

This is indicated by the **RUN** message (4) in the footer.

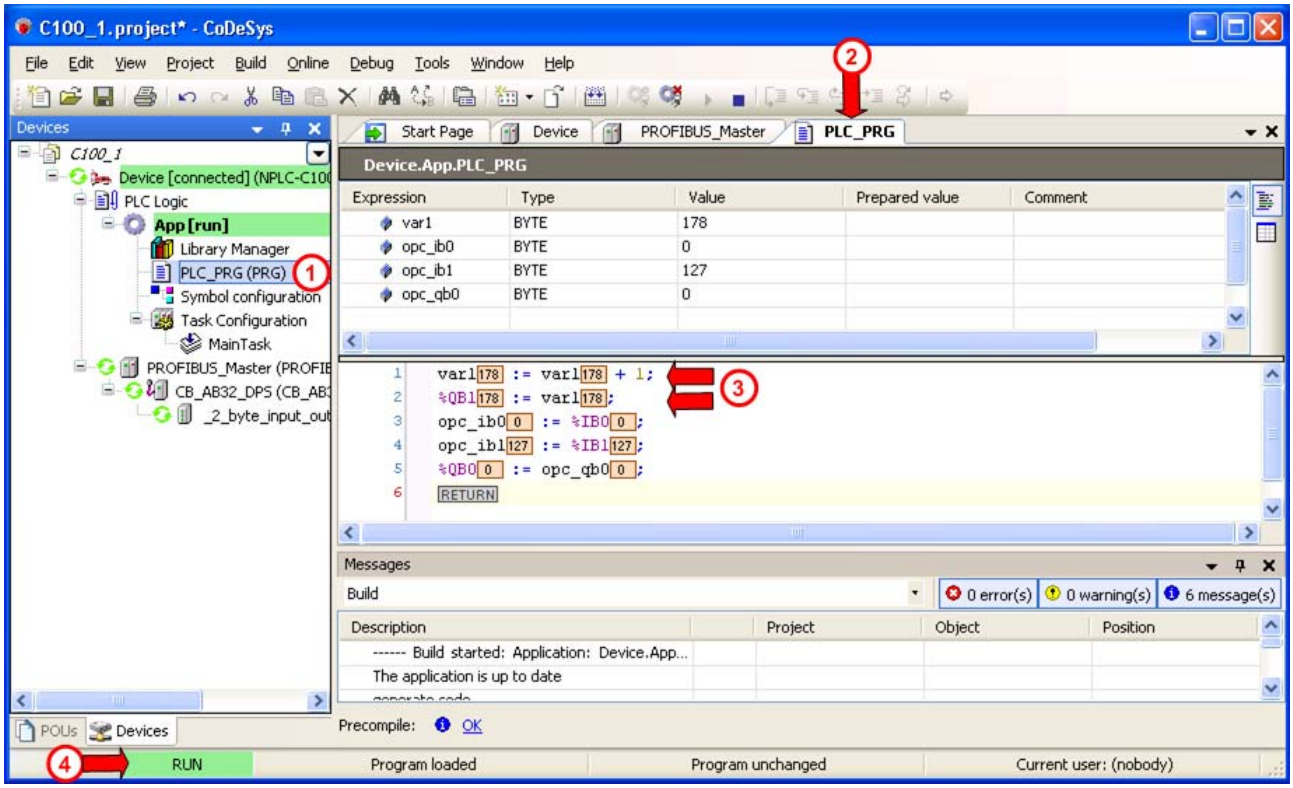


Figure 39: PLC Program Started

➢ In the **Devices** tree, double click **PLC_PRG** (1) or choose **PLC_PRG** tab (2) in the main window.

➢ In the indicated lines (see position (3) in the figure above), you can watch the variables being incremented.

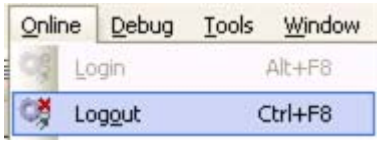
The PLC program increments in each PLC cycle the output byte %QB1 by one. Check, if the output data at the PROFIBUS DP Slave changes. Change the input data of the PROFIBUS DP Slave device and check, if this data reaches the PLC.

© Hilscher, 2009-2014

4.9 Disconnect from the PLC

Log out.

➢ In the menu, select
Online > Logout Ctrl+F8
to log out.



➢ The connection to the PLC is disconnected.

© Hilscher, 2009-2014

5 Visualizing

Data can be read from and written to the PLC from a visualization software with OPC interface (called OPC client in the following sections) via the OPC server.

For this, a configuration is necessary. The following sections describe the configuration steps to connect a visualization software via OPC to the PLC.

Requirements

Operational PLC, which means:

- installed hardware
- installed device driver: cifX Device Driver
- installed and operational server: netPLC CoDeSys Server
- the slot PLC has already been programmed with CoDeSys
- operational PROFIBUS DP slave device(s), which is/are connected via a PROFIBUS cable to the netPLC component.

Installed CoDeSys OPC Server

- Installed 3S OPC Server

The netPLC CoDeSys server on the PC must be started and must have access to the netPLC component.

© Hilscher, 2009-2014

5.1 OPC Configurator

1. Start OPC Configurator.

➢ Start the OPC Configurator: **Start > All Programs > 3S CoDeSys > CoDeSys OPC Server 3 > OPC Configurator**.

➢ The window of the OPC Configurator appears:

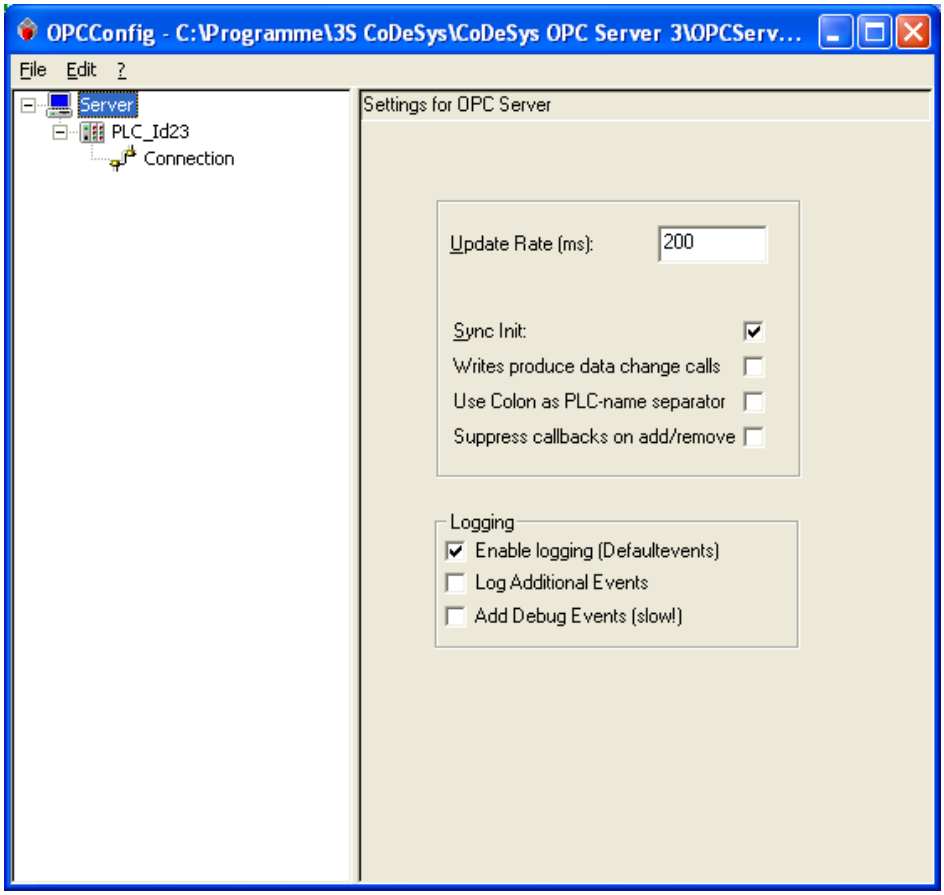


Figure 40: OPC Server Configuration

➢ Use the context menu to delete the **PLC_Id23** entry.

2. Append PLC.

➢ In the menu, choose **Edit > Append PLC** or use shortcut **Strg+A** on the keyboard.

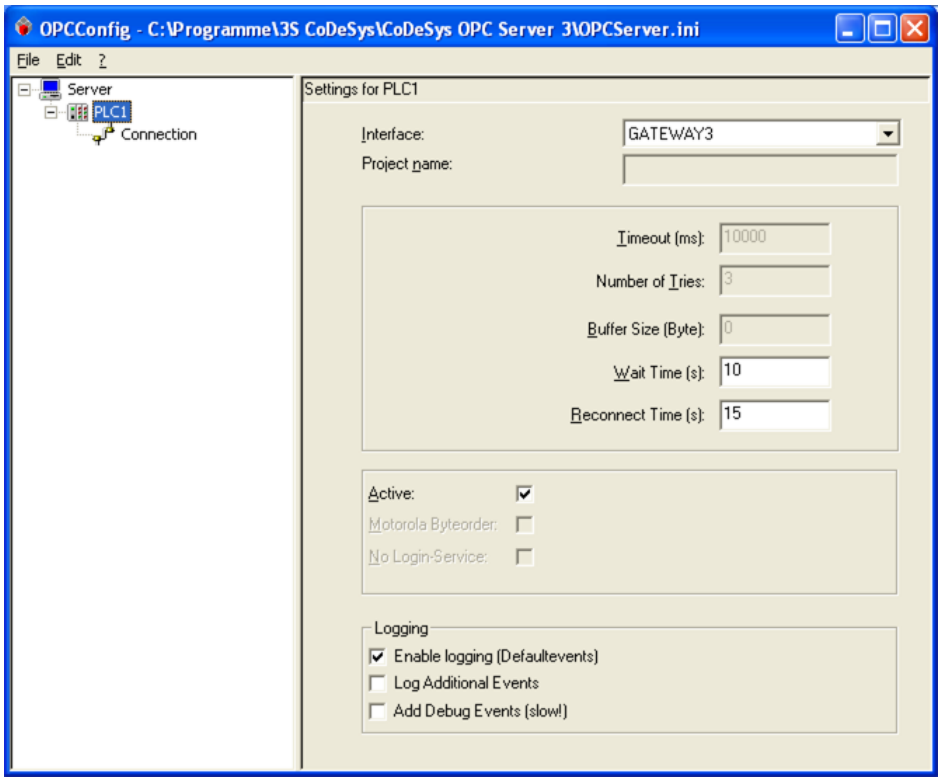
➢ The **Settings for PLC window** appears.

➢ The default name is **PLC1**.

© Hilscher, 2009-2014

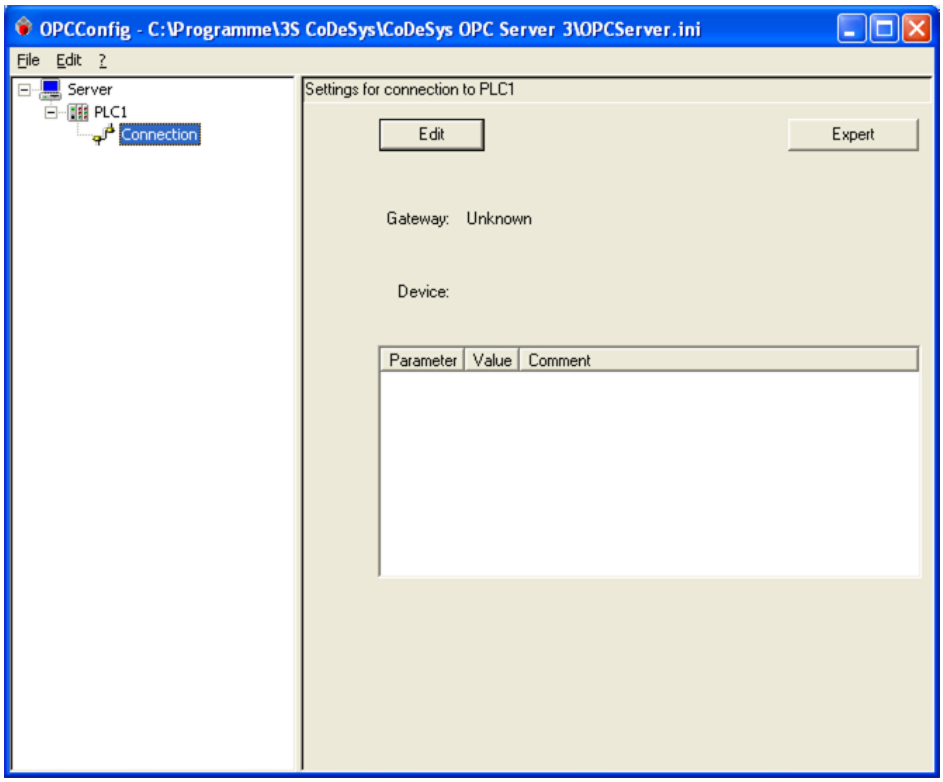
3. Set interface to GATEWAY3.

➢ In the **Interface** field, select **GATEWAY3** from the dropdown menu.



4. Edit Connection.

➢ In the tree, select **Connection** entry.



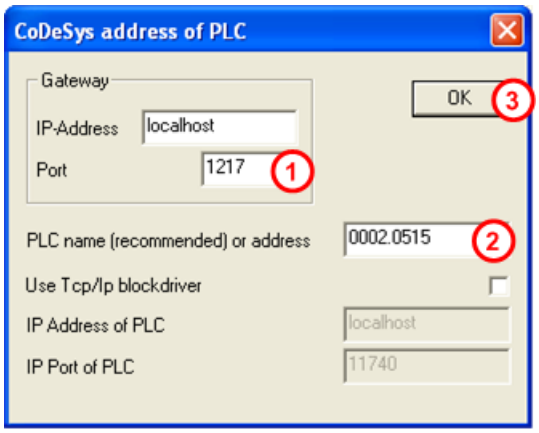
© Hilscher, 2009-2014

5. Edit settings.

➢ Click **Edit** button.

➧ The window to enter the address appears.

6. Enter CoDeSys PLC address.



➢ Use 1217 for **Port** (1).

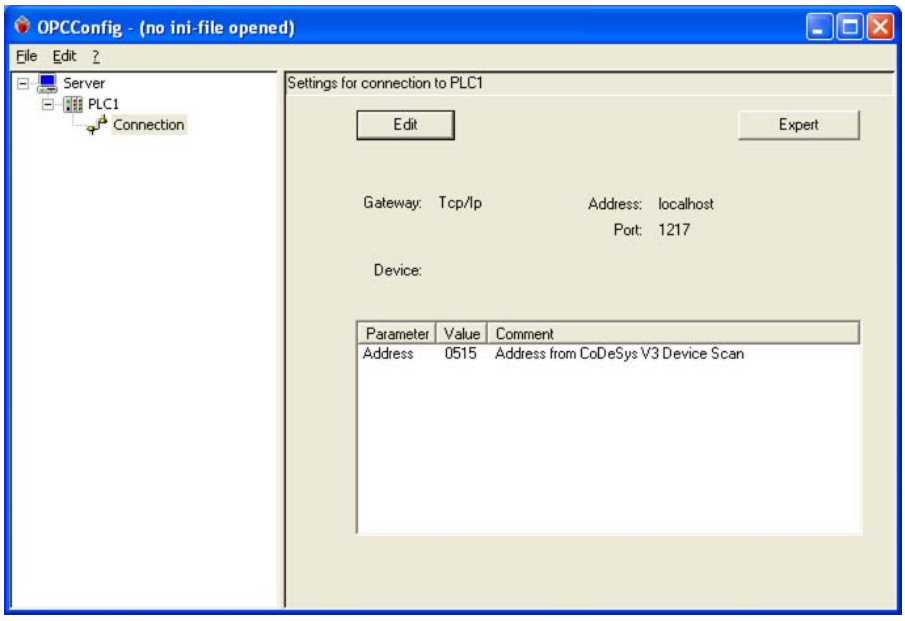
➢ In the **PLC name or address** field (2), enter the address by which the 3S Gateway can access the NPLC-C100 slot PLC, e. g. 0002.0515.

Important: Use the address shown in the CoDeSys programming software when the **Active Path** is set. See position (2) in *Figure 37: Path Set Active* on page 50.

7. Close address window.

➢ Click **OK** button (3) to close the window.

➧ The settings are stored temporarily.

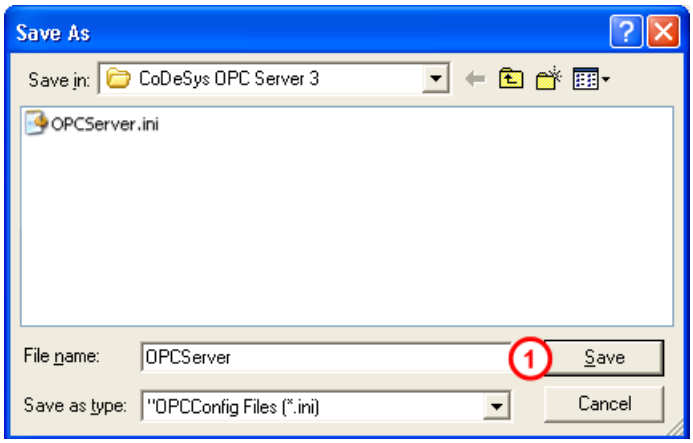


© Hilscher, 2009-2014

8. Save configuration.

➢ In the menu, choose **File > Save As**.

➢ Enter the file name **OPCServer.ini** and overwrite the existing file.



➢ Click **Save** button (1).

➢ The OPC configuration file is stored.

9. Exit OPC Configurator.

➢ In the menu, choose **File > Exit** to end the OPC configurator.

© Hilscher, 2009-2014

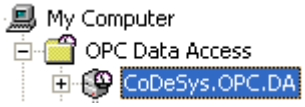
5.2 OPC Client

The access from an OPC client can only be sketched out here, because this very much depends on the visualization software you are using.

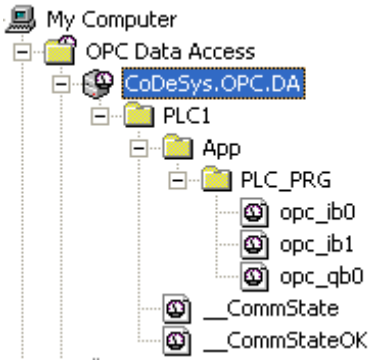
1. Start the OPC client software.

2. Establish a connection to the OPC server.

The CoDeSys OPC server starts automatically as background process on the target PC, on which the NPLC-C100 slot PLC is installed.



3. The variables are being read in, and can be browsed.



4. Add a group at the OPC client.

5. Add the items **opc_ib0**, **opc_ib1** and **opc_qb0** to this group.

6. Values are being read by the control unit.

7. Values can be written to the control unit: **opc_qb0** can be written.

Name	Value	Quality
PLC1.App.PLC_PRG.opc_ib0	0 (VT_UI1)	Good - Non-spe...
PLC1.App.PLC_PRG.opc_ib1	255 (VT_UI1)	Good - Non-spe...
PLC1.App.PLC_PRG.opc_qb0	13 (VT_UI1)	Good - Non-spe...

© Hilscher, 2009-2014

6 Host I/O Interface

The application program on the host can directly access input and output data of the PLC when using the slot PLC NPLC-C100. Therefore the **Host I/O Interface** has to be added to the CoDeSys project. This procedure is described in section *Add Host I/Os to NPLC-C100 Slot PLC* on page 19. The application program uses functions of the cifX Device Driver to read input data and to write output data.

The following figure shows the general structure.

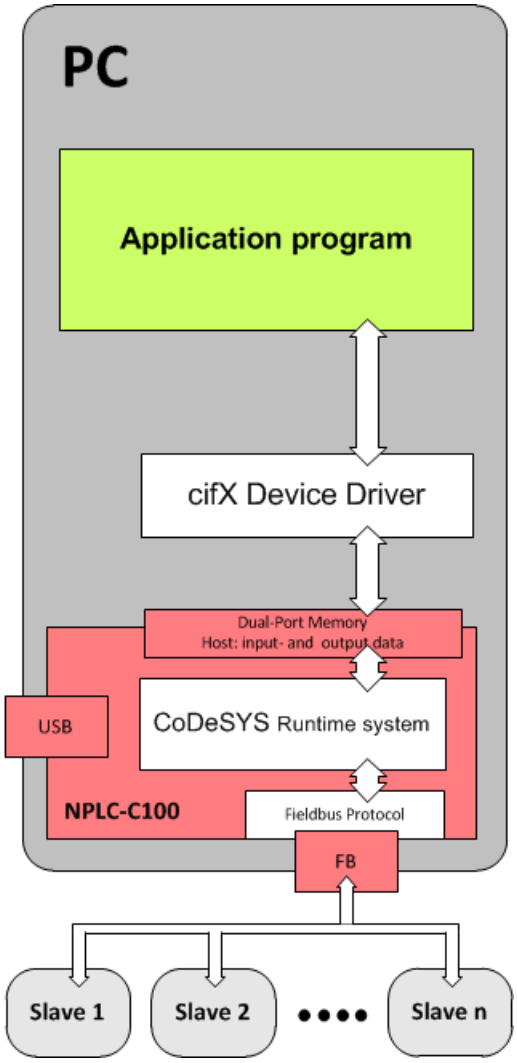


Figure 41: Host I/O Interface

The **Host I/O Interface** uses addresses %IB0 … %IB511 and %QB0 … %QB511 in the PLC memory, when the **Host I/O Interface** is the first ‚device' in the CoDeSys project (and slave devices are behind the Host I/O interface).

The **Host I/O Interface** uses the next free addresses in the PLC memory, when the **Host I/O Interface** is inserted in the CoDeSys project behind the slave devices. Example: When addresses %IB0 … %IB15 and %QB0 … %QB15 are already used for slave devices, the Host I/O Interface then uses addresses %IB16 … %IB527 and %QB16 … %QB527.

The input and output data in the host memory are always for the cifX Device Driver at offsets 0 … 511.

© Hilscher, 2009-2014

7 Functions

7.1 Add Functions Library

Before you can use the functions described in this section, you first have to add them to the functions library of CoDeSys.

To do so, proceed as follows:

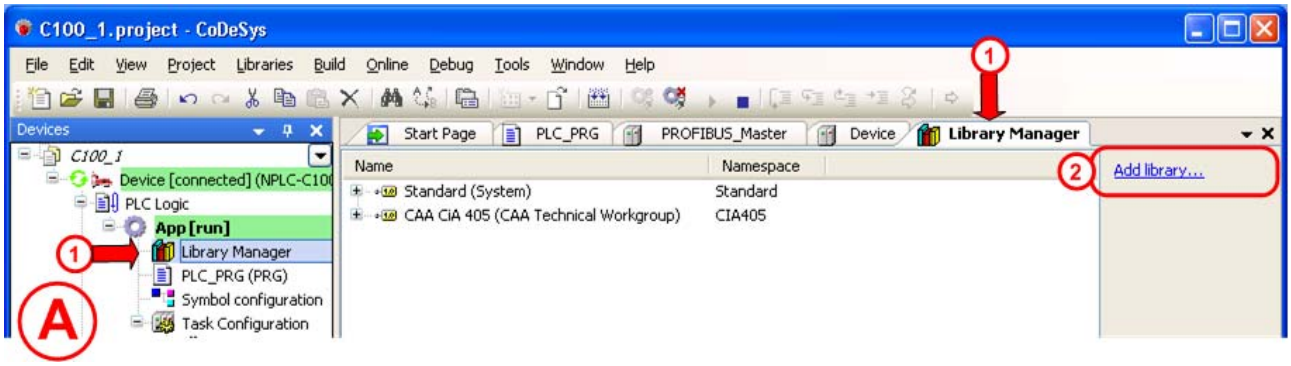
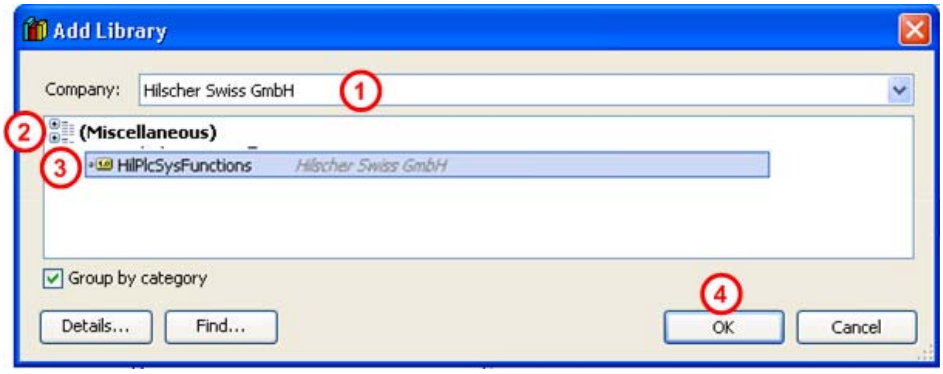


Figure 42: Add Library

1. In the **Devices** tree (A), double click **Library Manager** entry (1).

➢ In the **Library Manager**, click **Add library** (2).

➢ The **Add Library** window opens:



2. In the **Company** field (1), select **Hilscher Swiss GmbH** from the drop-down list.

➢ Click the plus symbol in front of the **(Miscellaneous)** category (2) to display subordinate entries.

➢ Select **HilPlcSysFunctions** entry (3).

➢ Click **OK** button (4).

➢ The library has been added to the **Library Manager**:

© Hilscher, 2009-2014

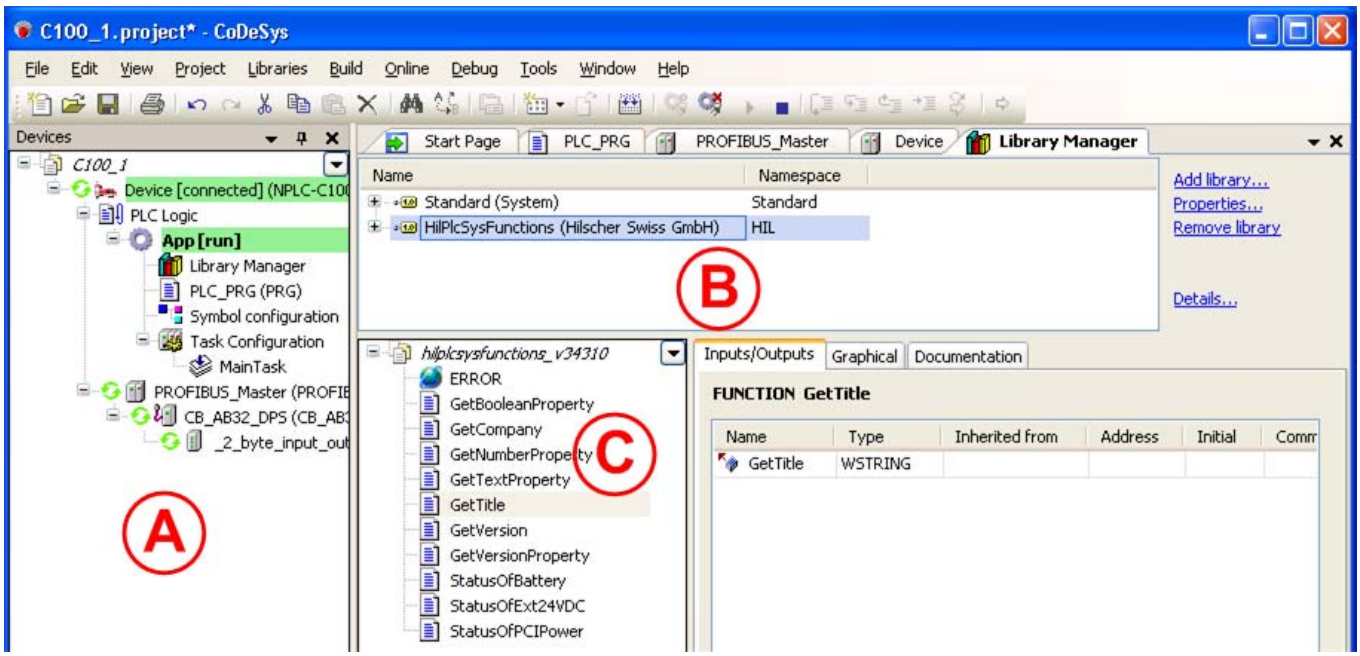


Figure 43: Add Library

The upper area Ⓑ of the Library Manager displays the integrated libraries, the lower area Ⓒ lists the functions belonging to the library selected in the upper area Ⓑ.

7.2 Functions – HilPlcSysFunctions

The function library HilPlcSysFunctions supports three functions for the netPLC components:

System function name	Function
StatusOfBattery	Checks the status of the battery and measures the battery voltage level.
StatusOfExt24V	Checks, if the netPLC component is supplied by an external power supply.
StatusOfPCIPower	Checks, if the netPLC component is supplied via PCI power.

Table 8: System Functions – HilPlcSysFunctions

To insert these functions into the PLC program, the input assistant can be used.

To open the input assistant, proceed as follows:

- Press F2

  or

- In the menu, choose **Edit > Input Assistant…**

  or

- In the editor, choose from the context menu the **Input Assistant** entry.

→ The **Input Assistant** opens:

© Hilscher, 2009-2014

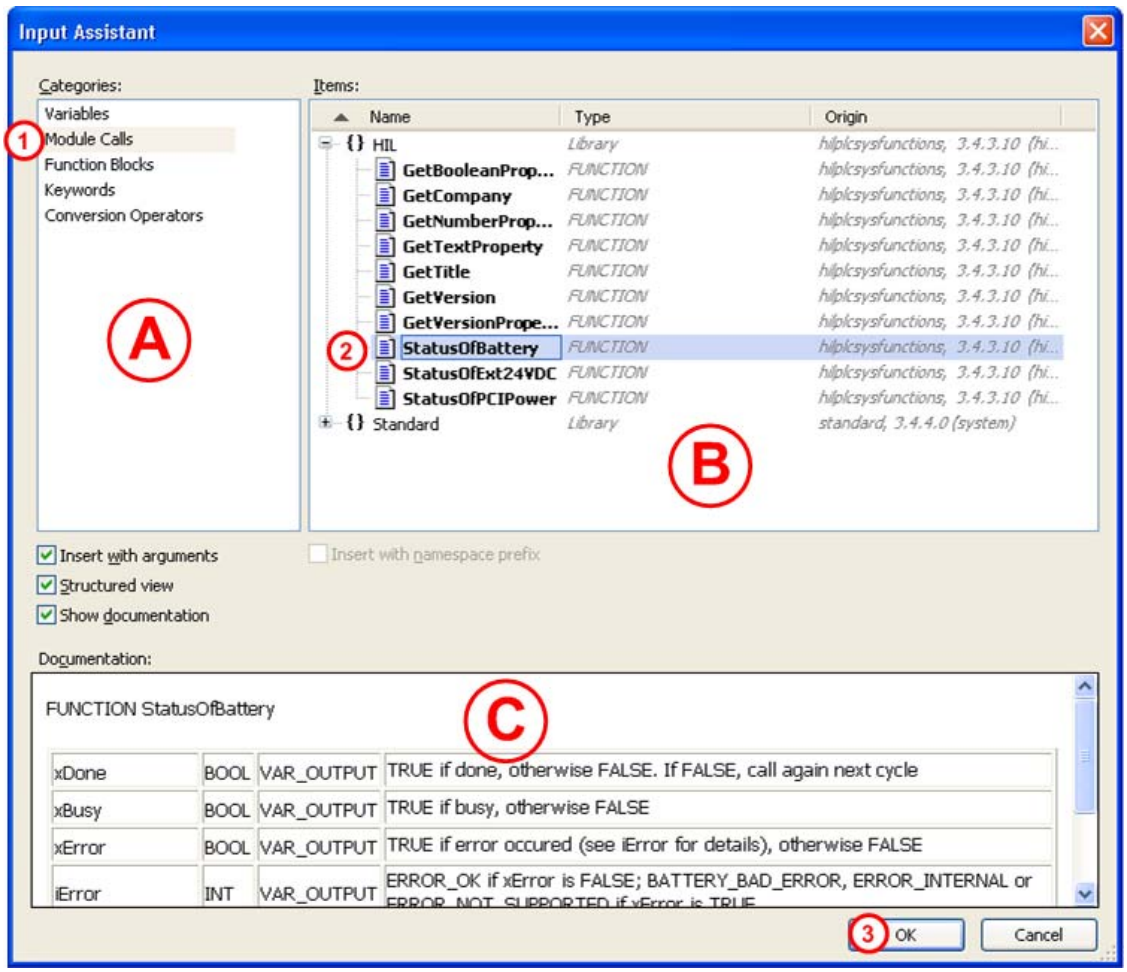


Figure 44: Input Assistant - Module Calls

- In the **Categories** list (A), select **Module Calls** entry (1).
- In the **Items** list (B), select, e. g., **StatusOfBattery** entry (2).
- In the documentation area of the window (C), the function parameters are displayed.
- Click **OK** button (3) to add the function to the PLC code.

© Hilscher, 2009-2014

7.2.1 Battery Status – StatusOfBattery

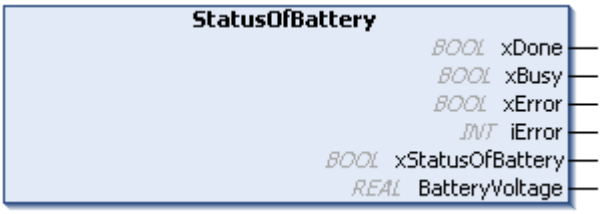


Figure 45: System Function StatusOfBattery

Function	**StatusOfBattery**			
Description	Checks for the status of the battery and measures the battery voltage level. Attention: During the check, the battery will be unloaded for a very short time, i. e. a discharge takes place. This is the reason why this function should not be called periodically, but from time to time only. The A/D converter used for this operation needs a certain amount of time for the act of measuring. Therefore, a State Machine is required for this function. See *Monitor netPLC C100/M100 Battery* section on page 71.			
Name	**Data type**	**In-/Output**	**Default**	**Comment**
xDone	BOOL	Output	FALSE	TRUE: System function ready FALSE: This function will be called in the next PLC cycle again
xBusy	BOOL	Output	FALSE	TRUE: System function in process FALSE: System function not in process
xError	BOOL	Output	FALSE	TRUE: Error occurred! The error code will be saved in iError FALSE: no error occurred
iError	INT	Output	0	In case xError has the value FALSE , then NO_ERROR: No Error (0) In case xError has the value TRUE, then BATTERY_BAD_ERROR: (1001) INTERNAL_ERROR: (1000) NOT_SUPPORTED: (1010) (Measuring not supported by hardware)
xStatusOfBattery	BOOL	Output	FALSE	TRUE: Battery status “Good” FALSE: Battery status “Bad” (Battery voltage < 2.75V) → Change battery
BatteryVoltage	REAL	Output	0	Measured voltage level of the battery in Volts. A battery voltage level above 2.75 V means that the battery voltage is sufficient.

Table 9: System Function StatusOfBattery

© Hilscher, 2009-2014

7.2.1.1 Effects of the Function in netPLC Components not equipped with Hardware for Battery Voltage Monitoring

In these components, the function will not cause fatal error.

The function will be returned with the following values:

Name	Data type	In-/Output	Default	Comment
xDone	BOOL	Output	FALSE	TRUE: System function ready
xBusy	BOOL	Output	FALSE	FALSE: System function not in process
xError	BOOL	Output	FALSE	TRUE: Error occurred! The error code will be saved in iError.
iError	INT	Output	0	NOT_SUPPORTED: (1010) (Measuring not supported by hardware)
xStatusOfBattery	BOOL	Output	FALSE	TRUE: Battery status “Good”
BatteryVoltage	REAL	Output	0	3 V

Table 10: System Function StatusOfBattery, Hardware without AD Converter

© Hilscher, 2009-2014

7.2.2 Status External Voltage Supply – StatusOfExt24VDC

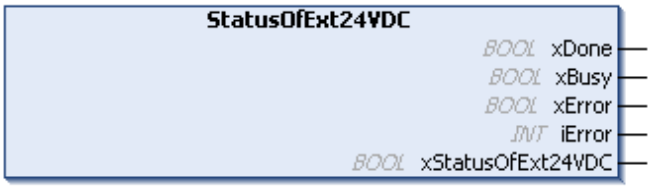


Figure 46: System function StatusOfExt24VDC

Function	**StatusOfExt24VDC**			
Description	Checks, if the PLC is supplied by an external power supply.			
Name	**Data type**	**In-/Output**	**Default**	**Comment**
xDone	BOOL	Output	FALSE	TRUE: System function ready FALSE: This function will be called in the next PLC cycle again
xBusy	BOOL	Output	FALSE	TRUE: System function in process FALSE: System function not in process
xError	BOOL	Output	FALSE	TRUE: Error occurred! The error code will be saved in iError FALSE: no error occurred
iError	INT	Output	0	In case xError has the value FALSE , then NO_ERROR: No Error (0) In case xError has the value TRUE, then INTERNAL_ERROR: (1000) NOT_SUPPORTED: (1010)
xStatusOfExt24VDC	BOOL	Output	FALSE	TRUE: External 24 V power supply available Uext. > 18 V FALSE: External 24 V power supply not available Uext. < 18 V

Table 11: System Function StatusOfExt24VDC

© Hilscher, 2009-2014

7.2.3 Status PCI Power Supply – StatusOfPCIPower

Figure 47: System Function StatusOfPCIPower

Function	**StatusOfPCIPower**			
Description	Checks, if the slot PLC is supplied with the PCI power.			
Name	**Data type**	**In-/Output**	**Default**	**Comment**
xDone	BOOL	Output	FALSE	TRUE: System function ready FALSE: This function will be called in the next PLC cycle again
xBusy	BOOL	Output	FALSE	TRUE: System function in process FALSE: System function not in process
xError	BOOL	Output	FALSE	TRUE: Error occurred! The error code will be saved in iError FALSE: no error occurred
iError	INT	Output	0	In case xError has the value FALSE, then NO_ERROR: No Error (0) In case xError has the value TRUE, then INTERNAL_ERROR: (1000) NOT_SUPPORTED: (1010)
xStatusOfPciPower	BOOL	Output	FALSE	TRUE: PCI power supply available FALSE: PCI power supply not available

Table 12: System Function StatusOfPCIPower

© Hilscher, 2009-2014

7.3 Reset Behavior

The slot PLC has no connection via the PCI bus to the PC reset signal. This means that a reset of the PC for example when using the front panel reset button of the computer does not influence the slot PLC behavior.

The PowerOn reset will be served from both power sources (internal und external).

7.4 RUN/STOP Switch

The PLC program's operating mode in the netPLC component can be changed using the physical switch at the front of the netPLC component:

Position	Function
RUN	Cyclic running of loaded PLC program.
STOP	Cyclic running of loaded PLC program is stopped. Loading a new PLC program or updating the firmware is possible.

Table 13: RUN/STOP Switch

In case the CoDeSys programming interface is logged in, you can change the operating condition within CoDeSys without using the physical switch.

Please pay attention: This may result in a condition where CoDeSys programming interface switches the PLC into STOP mode while the physical switch is still in position RUN. Thus, during the next boot up cycle, the PLC will automatically go into the RUN mode, even if this might not be intended by the user. Please move the switch into position STOP to prevent an unintended restart.

Switch position has priority over the software control!

© Hilscher, 2009-2014

7.5 Monitor netPLC C100/M100 Battery

The battery voltage of the hardware components NPLC-C100 and NPLC-M100 can be monitored with the **StatusOfBattery** system function. For further information, see *Battery Status – StatusOfBattery* section on page 66.

Example of a code for this function:

```
(* a rising edge on S1.1 on the M100 evaluation board is used for triggering the measure-
ment *)
fTriggerState := %IX9.0;
IF fTriggerState AND NOT fLastTriggerState THEN
     fMeasurementEnabled := TRUE;
     iMeasurementCycles := 0;
END_IF
fLastTriggerState := fTriggerState;

IF fMeasurementEnabled THEN
     iMeasurementCycles := iMeasurementCycles + 1;
     (* check external 24 VDC power supply *)
     StatusOfExt24VDC (xDone=> fExt24VDone, xBusy=> fExt24VBusy, xError=> fExt24VError,
iError=> iExt24VError, xStatusOfExt24VDC=> fExt24VState);
     (* check 3.3 VDC power supply from PCI bus *)
     StatusOfPCIPower (xDone=> fPci3V3Done, xBusy=> fPci3V3Busy, xError=> fPci3V3Error,
iError=> iPci3V3Error, xStatusOfPciPower=> fPci3V3State);
     (* check battery voltage *)
     StatusOfBattery (xDone=> fBatteryDone, xBusy=> fBatteryBusy, xError=> fBatteryError,
iError=> iBatteryError, xStatusOfBattery=> fBatteryState, BatteryVoltage=>
rBatteryVoltage);
     IF fBatteryDone OR fBatteryError THEN
          fMeasurementEnabled := FALSE;
     END_IF
END_IF
```


© Hilscher, 2009-2014

8 Bus Diagnostic

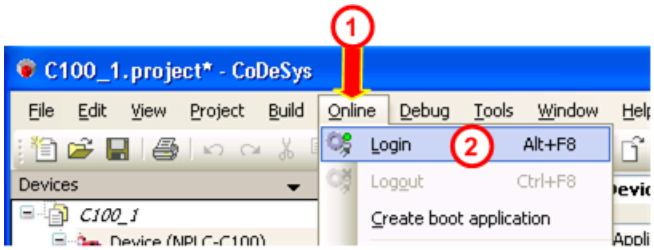
8.1 Start Up and Test the Bus

Note: By downloading the PLC program to the PLC component, the bus configuration is also downloaded.

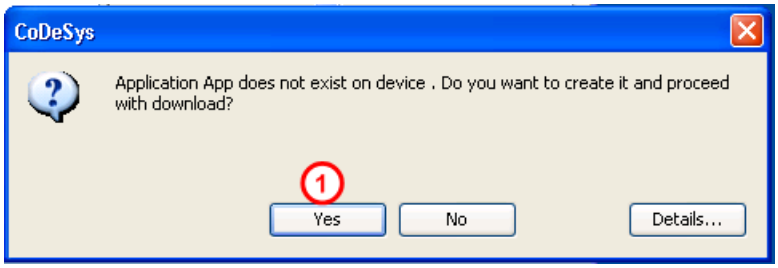
8.1.1 Login and Download Program

1. Login to the PLC.

➢ In the menu, choose **Online > Login**.



↬ The software verifies, if the project in the PLC is the same as the project on the PC. If no PLC program is in the PLC, then the message “Application App does not exist on device. Do you want to create it and proceed with download?“ appears. If the projects are different, then the message “Unknown version of application App on target. Do you want to perform a download and replace the application?” appears.



2. Start Download.

➢ Answer the question “Application App does not exist on device. Do you want to create it and proceed with download?“ with **Yes** (1).

↬ The application is downloaded to the PLC.

↬ With the start of the download, the PLC goes into STOP mode.

© Hilscher, 2009-2014

8.1.2 Device Assignment

1. Open **Devices** tree.

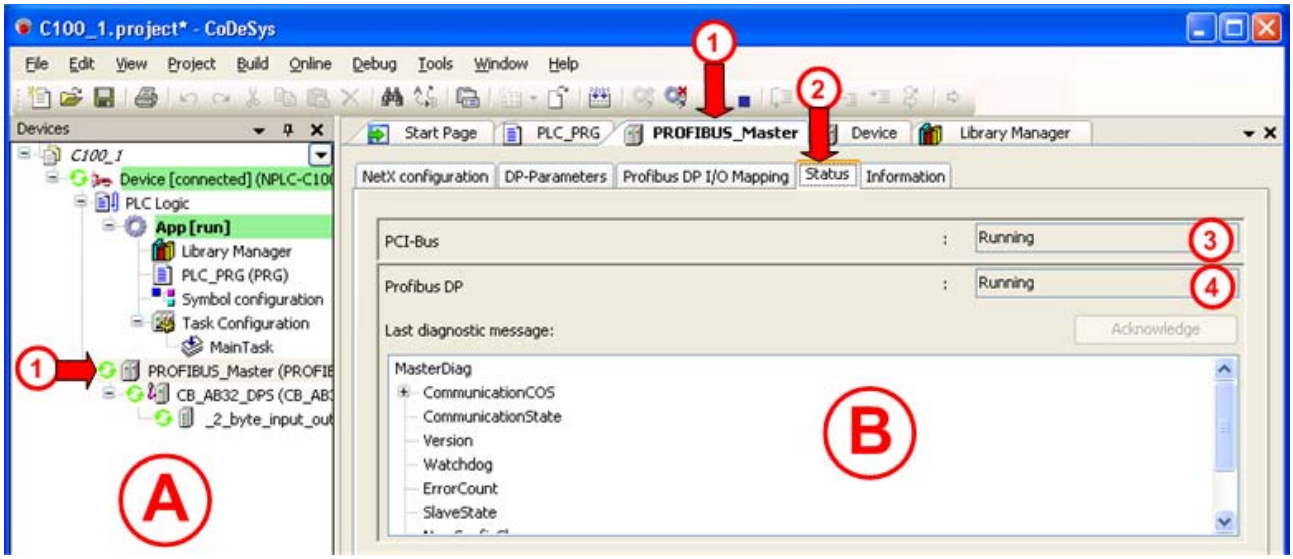


Figure 48: Bus Diagnostic (1) PROFIBUS Example

- In the **Devices** tree (A), double click bus master entry (1) to open the Fieldbus information.
- Click **Status** tab (2) to open the bus information.
- If no errors have been diagnosed on the bus system, the tab looks as shown in the figure above.

  The field indicated at position (3) provides information about the PCI Bus (if it is a netPLC C100 component).

  The field indicated at position (4) provides information about the Fieldbus.

In case of error, the tab looks as shown in the figure below:

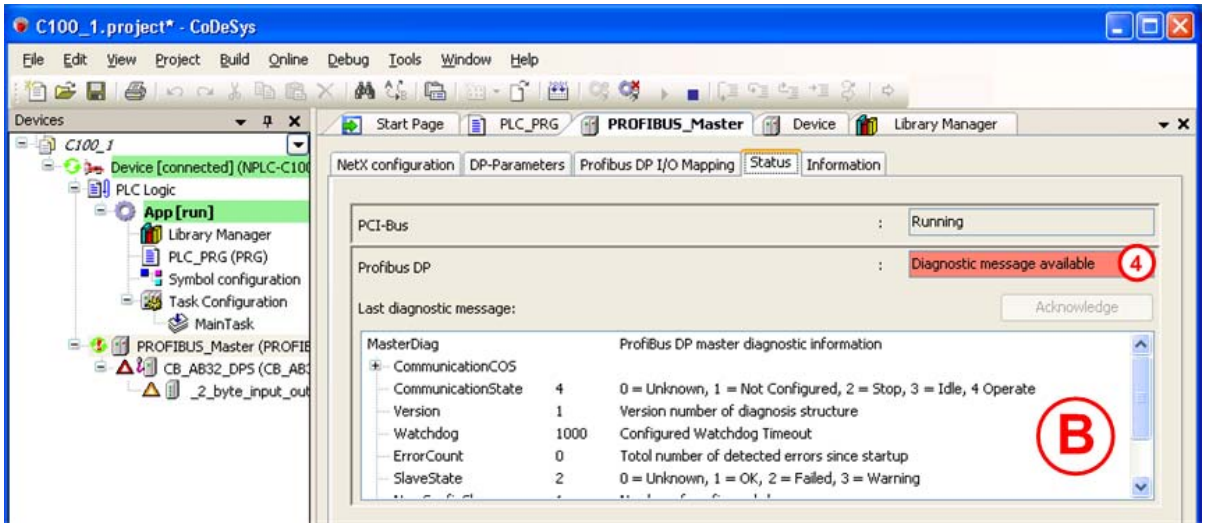


Figure 49: Bus Diagnostics (2) PROFIBUS Example

© Hilscher, 2009-2014

If errors have been diagnosed on the Fieldbus system:

The field indicated at position ④ informs you, that diagnostic messages are available. These messages are listed in the main area Ⓑ of the tab.

8.1.3 Verify Input and Output Data

1. Verify the input and output data by using the I/O Mapping tab.

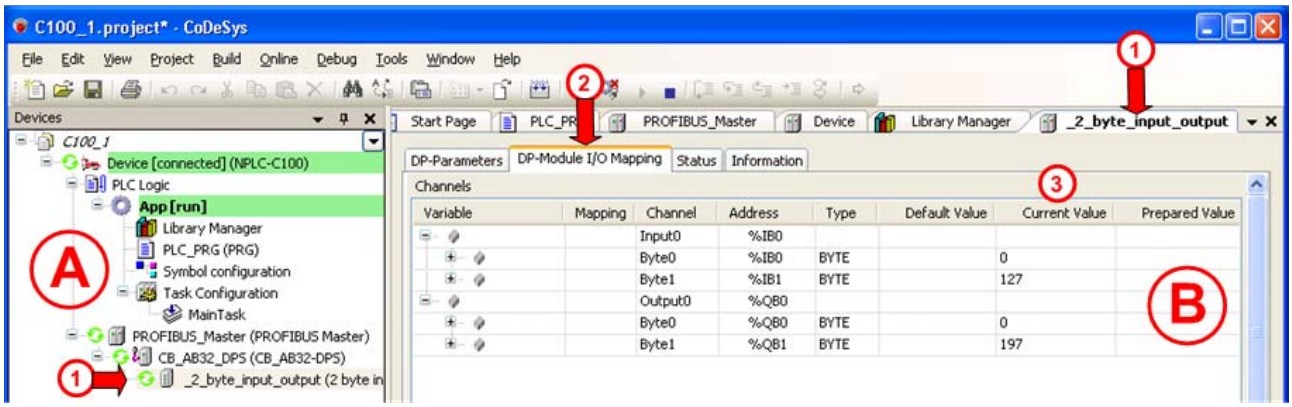


Figure 50: Bus Diagnostic (3) Slave I/O Data

➢ In the Devices tree Ⓐ, select slave I/O Data entry ①.

➢ In the main area Ⓑ, select **DP-Module I/O Mapping** tab ②.

In the main area Ⓑ, the inputs and outputs of the slave are shown. The current values of the signals are shown in the **Current Value** column ③

© Hilscher, 2009-2014

9 Troubleshooting

General

- Check, if the requirements for the operation of the netPLC component are fulfilled.

For further information on this, please refer to the hardware documentation of the corresponding netPLC component.

Cable

- Check the used cable for correct pinning.

Configuration

- Check, if the master configuration matches the slave configuration, e. g. with regard to bus address and Baud rate.

Use the status information of the individual device components in the Devices tree of the project.

Use the **I/O Mapping** tab for the I/O Signals of the Slave components in the **Devices** tree.

Use the information provided by the LEDs of the devices.

9.1 Error Message During Boot Application Creation

The following error message might appear after you have created a symbol configuration while creating a boot application.

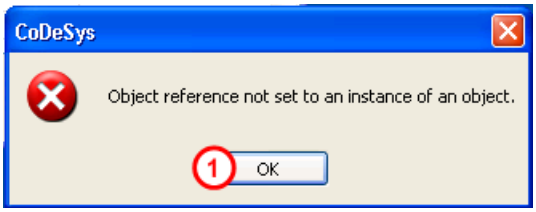


Figure 51: Object reference not set…

Proceed as follows:

1. Click **OK** button (1) to close pop-up window.
2. Save project.
3. Exit CoDeSys application.
4. Start CoDeSys again.
5. Load the project which you have previously saved.
6. Start creating boot application again.

© Hilscher, 2009-2014

10 LEDs

Bedeutung und Benennung der LEDs an der netPLC Baugruppe

LED-Benennung in der Geräte-zeichnung	CANopen Master	DeviceNet Master	PROFIBUS DP Master
SYS (gelb / grün / aus)	**SYS** (Systemstatus)	**SYS** (Systemstatus)	**SYS** (Systemstatus)
APL (gelb / grün)	**APL** (Application, *netPLC*)	**APL** (Application, *netPLC*)	**APL** (Application, *netPLC*)
COM 0 (Kommu-nikationsstatus) (rot / grün)	**-** -	**MNS** *(Kommuni-kationsstatus)*	**COM** *(Kommuni-kationsstatus)*
COM 1 (Kommu-nikationsstatus) (rot / grün)	(*nicht verwendet*)	(*nicht verwendet*)	(*nicht verwendet*)

Table 14: LED-Benennung in der Gerätezeichnung und LED-Bedeutung und -Benennung nach Protokoll (Feldbus-Systeme)

10.1 LED SYS

The following table describes the meaning of the system LED for netPLC components.

LED	Color	State	Meaning
SYS	**Duo LED yellow/green**		
	(green)	On	Operating System running
	(green/yellow)	Blinking green/yellow	Bootloader is waiting for firmware
	(yellow)	Blinking	Heavy system error: PLC in stop.
	(yellow)	static	Bootloader is waiting for software
	(yellow/green/ off)	Continously blinking	SecMem not readable.
	-	Off	Power supply for the device is missing or hardware defect.

Table 15: System LED

© Hilscher, 2009-2014

10.2 LED APL

LED	Farbe	Zustand	Bedeutung
APL	**Duo LED grün/gelb**		
	(grün)	Ein	SPS im Zustand RUN. SPS-Programm wird ausgeführt.
	(grün)	Zyklisch blinkend 1 Hz	Kein bootbares SPS-Programm vorhanden.
	(gelb)	Ein	SPS im Zustand STOP. SPS-Programm wird nicht ausgeführt.
	(gelb)	Zyklisch blinkend 4 Hz	Unerwarteter Fehler bei der Ausführung des SPS-Programms aufgetreten.

Tabelle 1: LED APL CoDeSys

10.3 LED CANopen Master

The LED for CANopen master is not used.

10.4 LED DeviceNet Master

The subsequent table describes the meaning of the LEDs for the fieldbus when the firmware of the DeviceNet Master protocol is loaded to the device.

LED	Color	State	Meaning
MNS	**Duo LED red/green**		
	(green)	On	Device is online and has established one or more connections
	(green)	Flashing	Device is online and has established no connection
	(green/red/ off)	Green/Red/ Off	Selftest after power on: Green on for 0,25 s, then red on for 0,25 s, then off
	(red)	On	Critical connection failure; device has detected a network error: duplicate MAC-ID or severe error in CAN network (CAN-bus off)
	(red)	Flashing	Connection timeout
	(off)	Off	After start of the device and during duplicate MAC-ID check

Table 16: LEDs DeviceNet Master

© Hilscher, 2009-2014

10.5 LED PROFIBUS DP-Master

Name COM0 in the device drawing.

The following table describes the meaning of the LED when the firmware of the PROFIBUS DP Master protocol is loaded to the device.

LED	Color	State	Meaning
COM	**Duo LED red/green**		
	(green)	Flashing acyclic	No configuration or stack error
	(green)	Flashing cyclic	Profibus is configured, but bus communication is not yet released from the application
	(green)	On	Communication to all Slaves is established
	(red)	Flashing cyclic	Communication to at least one Slave is disconnected
	(red)	On	Communication to one/all Slaves is disconnected or annother serious error has occured. Redundant Mode: The active Master was not found.

Table 17: LEDs PROFIBUS DP Master

© Hilscher, 2009-2014

11 Error Codes

11.1 Error Codes Functions HilPlcSysFunctions

Error codes are of data type INT.

Error code	Name Description
0	NO_ERROR No error occurred
1000	INTERNAL_ERROR Unexpected internal error has occurred
1001	BATTERY_BAD_ERROR The status of the battery is “bad”. The battery has to be changed.
1010	FCT_NOT_SUPPORTED_ERROR This function is not supported.

Table 18: Error Codes Functions HilPlcSysFunctions_netPLC

© Hilscher, 2009-2014

12 Technical Data Protocols

12.1 CANopen Master and DeviceNet Master

The same firmware file is used for both CANopen Master and DeviceNet Master. Communication functions of CANopen are running (as IEC code) in CoDeSys Control.

CANopen Master

Parameter	Description
Maximum number of CANopen nodes	126
Maximum number of cyclic input data	3584 bytes Up to 512 bytes from them can be used for Host input data for NPLC-C100.
Maximum number of cyclic output data	3584 bytes Up to 512 bytes from them can be used for Host output data for NPLC-C100.
Maximale Anzahl empfangener PDOs	512
Maximale Anzahl übertragener PDOs	512
Exchange of process data	Via PDO transfer: - synchronized, - remotely requested and - event driven (change of date)
Functions	Emergency message (consumer and producer) Node guarding / life guarding, heartbeat PDO mapping NMT Master SYNC protocol (producer)
Baud rates	10 kBits/s, 20 kBits/s, 50 kBits/s, 100 kBits/s, 125 kBits/s, 250 kBits/s, 500 kBits/s, 800 kBits/s, 1 MBits/s
Data transport layer	CAN Frames
CAN Frame type for CANopen	11 Bit
Reference to firmware/stack version	V4.0.x.x

Table 19: Technical Data CANopen Master Protocol

© Hilscher, 2009-2014

DeviceNet Master

Parameter	Description
Maximum number of DeviceNet slaves	63
Maximum number of total cyclic input data	3584 bytes Up to 512 bytes from them can be used for Host input data for NPLC-C100.
Maximum number of total cyclic output data	3584 bytes Up to 512 bytes from them can be used for Host output data for NPLC-C100.
Maximum number of cyclic input data	255 bytes/connection
Maximum number of cyclic output data	255 bytes/connection
Maximum Configuration data	1000 bytes/slave
Connections	Bit Strobe Change of State Cyclic Poll Explicit Peer-to-Peer Messaging (for paramerization only)
Function	Quick Connect
Fragmentation	Explicit and I/O
UCMM	Supported
Objects	Identity Object (Class Code 0x01) Message Router Object (Class Code 0x02) DeviceNet Object (Class Code 0x03) Connection Object (Class Code 0x05) Acknowledge Handler Object (Class Code 0x06)
Baud rates	125 kBits/s, 250 kBit/s, 500 kBit/s Auto baudrate detection is not supported
Data transport layer	CAN frames
Limitations	User data transfer through the gateway only via IO connections
Reference to firmware/stack version	V4.0.x.x/V2.3.x.x

Table 20: Technical Data DeviceNet Master Protocol

© Hilscher, 2009-2014

12.2 PROFIBUS DP-Master

Parameter	Description
Maximum number of PROFIBUS DP slaves	125
Maximum number of total cyclic input data	3584 bytes Up to 512 bytes from them can be used for Host input data for NPLC-C100.
Maximum number of total cyclic output data	3584 bytes Up to 512 bytes from them can be used for Host output data for NPLC-C100.
Maximum number of cyclic input data	244 bytes/slave
Maximum number of cyclic output data	244 bytes/slave
Configuration data	Max. 244 bytes per slave
Parameterization data per slave	7 bytes/slave standard parameters Max. 237 bytes/slave application specific parameters
Acyclic communication	The following DPV1 class 1 services can be used with the CoDeSys library *IIoDrvDPV1C1*: - read/write - alarm and alarm ack
Baud rate	9,6 kBits/s, 19,2 kBits/s, 31,25 kBits/s, 45,45 kBits/s 93,75 kBits/s, 187,5 kBits/s, 500 kBits/s, 1, 5 MBits/s, 3 MBits/s, 6 MBits/s, 12 MBit/s Auto baudrate detection is not supported
Data transport layer	PROFIBUS FDL
Limitations	DPV1 services class 2 are not supported DPV2 services are not implemented
Reference to firmware/stack version	V4.0.x.x/V2.6.x.x

Table 21: Technical Data PROFIBUS DP Master Protocol

© Hilscher, 2009-2014

13 Appendix

13.1 netPLC CoDeSys Server Program

The netPLC CoDeSys server program is the link between the 3S gateway (and thus to the CoDeSys programming interface) on one side and the device driver (cifX device driver) on the other, to communicate finally with the netPLC component via PCI. The 3S gateway may be installed on the same PC as the PLC, or it may be installed on a remote machine (connected via the local Ethernet network).

13.1.1 Starting netPLC CoDeSys Server

The netPLC CoDeSys Server will be installed under autostart in windows, thus starting automatically each time the PC is powered.

To start the server manually, select **Start > All Programs > Hilscher GmbH > netPLC CoDeSys Server**.

In case the netPLC CoDeSys Server Program is started and running, you will find the following icon in the system tray:

Figure 52: netPLC CoDeSys Server Tray Icon

© Hilscher, 2009-2014

13.1.2 netPLC CoDeSys Server Window

Double click the CoDeSys server tray icon to open the **netPLC CoDeSys Server** window.

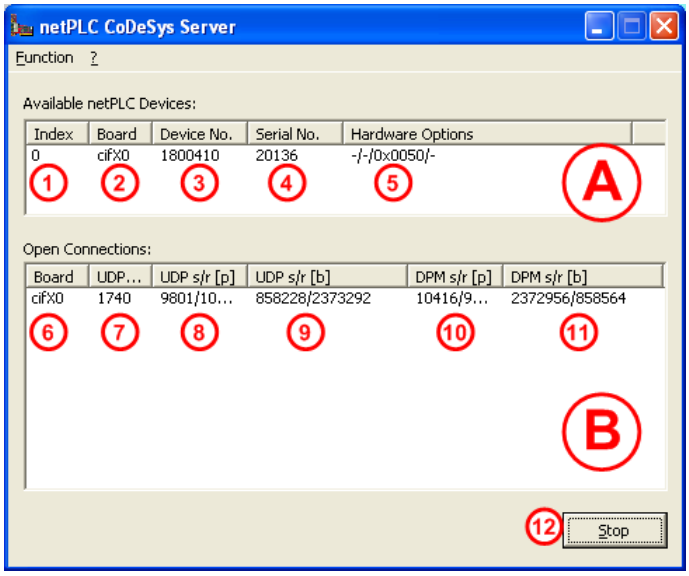


Figure 53: netPLC CoDeSys Server Window

The displayed items have the following meaning:

The upper area Ⓐ of the window displays the available components.

The lower area Ⓑ of the window displays the components to which a connection has been established.

Element	Description
① Index	Consecutive numbering
② Board	Shows which slot PLC NPLC-C100 (cifX0, cifX1, …) the netPLC CoDeSys Server is able to access via the device driver.
③ Device No.	Part number of the component
④ Serial No.	Serial number of the component
⑤ Hardware Options	Shows interface type (id)
⑥ Board	Shows to which slot PLC NPLC-C100 (cifX0, cifX1, …) access has been established via the PC card device driver.
⑦ UDP Port	Port, the netPLC CoDeSys Server is using to send and receive data packets from and to the 3S Gateway
⑧ UDP s/r [p]	Counter for transmitted packets (UDP s = send) as well as for received packets (UDP r = receive) via the connection to the 3S gateway
⑨ UDP s/r [b]	Counter for transmitted bytes (UDP s = send) as well as for received bytes (UDP r = receive) via the connection to the 3S gateway
⑩ DPM s/r [p]	Counter for transmitted packets (DPM s = send) as well as for received packets (DPM r = receive) via the device driver to the slot PLC.
⑪ DPM s/r [b]	Counter for transmitted bytes (DPM s = send) as well as for received bytes (DPM r = receive) via the device driver to the slot PLC.

Table 22: netPLC CoDeSys Server Windows – Items

© Hilscher, 2009-2014

13.1.3 Stopping netPLC CoDeSys Server

To quit operation of the netPLC CoDeSys Servers, click on the system tray icon with the right mouse button. From the context menu, select **Stop**, or press the button **Stop** ⑫ in the window of the server itself.

Answer the confirmation prompt with **Yes** to quit the netPLC CoDeSys Server.

© Hilscher, 2009-2014

13.2 Example Projects

The DVD features four example projects. Brief descriptions of these examples are provided in the subsequent sections.

To open, load and start a PLC example program, proceed as follows:

- In the CoDeSys menu, choose **File > Open Project** to open the project.
- Set active path.
- In the CoDeSys menu, choose **Online > Login to [Device: PLC Logic]** to load the program to the slot PLC (or press Alt + F8 on the keyboard).
- In the CoDeSys menu, choose **Online > Create boot application for App [Device: PLC Logic]**.
- In the CoDeSys menu, choose **Online > Start App [Device: PLC logic]** to start the PLC program (or press F5 on the keyboard).

13.2.1 netPLC System Function

Project name	netPLC System Functions.project
Folder	Supplements & Examples\CoDeSys Sample Projects\netPLC System Functions
Short description	The project consists of a netPLC NPLC-C100-DP.
PLC program	The PLC program shows the usage of the system functions StatusOfBattery, StatusOfExt24V and StatusOfPCIPower.

Table 23: Example – netPLC System Function

The system functions are described in *Add Functions Library* section on page 63.

© Hilscher, 2009-2014

13.2.2 netPLC Open Process Control (OPC)

Project name	Open Process Control.project
Folder	Supplements & Examples\CoDeSys Sample Projects\ netPLC Open Process Control (OPC)
Short description	The project consists of a netPLC NPLC-C100-DP.
PLC program	The PLC program shows how to create variables and how use the symbol configurator to make them accessible for the OPC server. From a remote OPC client it is possible to access the variables opc_qb0 value (0 .. 255) and to write a value. The PLC program copies this value of opc_qb0 variable to the Variable opc_ib0 and the OPC Client can read back this value from opc_ib0. The PLC program uses the same mechanism for the variables opc_qb1 and opc_ib1.

Table 24: Example – netPLC Open Process Control

To use this project, proceed as follows:

- In the CoDeSys menu, choose **File > Open Project** to open the project.
- Set active path.
- When the active path is set, remember the address, because you have to enter this address again in the OPC Configurator.
- In the CoDeSys menu, choose **Online > Login to [Device: PLC Logic]** to load the program to the netPLC C100 (or press Alt + F8 on the keyboard).
- In the CoDeSys menu, choose **Online > Create boot application for App [Device: PLC Logic]**.
- In the CoDeSys menu, choose **Online > Start App [Device: PLC logic]** to start the PLC program (or press F5 on the keyboard).
- Use the OPC Configurator to create an OPC configuration.
  This is described in the *OPC Configurator* section on page 57.
- Connect your OPC client to the OPC server.
- Browse items (display items).
- Select items opc_qb0, opc_qb1, opc_ib0 and opc_ib1.
- Write values between 0 to 255 to the items opc_qb0 and opc_qb1 and check read values for items opc_ib0 and opc_ib1.

13.2.3 netPLC Real Time Clock

Project name	netPLC Real Time Clock.project
Folder	Supplements & Examples\CoDeSys Sample Projects\netPLC Real Time Clock
Short description	The project consists of a netPLC NPLC-C100-DP.
PLC program	The PLC program shows how the functions SetDateAndTime and GetDateAndTime are used to change the date and time of the netPLC C100 and how to read the values back.

Table 25: Example – netPLC Real Time Clock

© Hilscher, 2009-2014

13.2.4 NPLC-C100-DP PROFIBUS Simple IO

Project name	NPLC-C100-DP Profibus Simple IO.project
Folder	Supplements & Examples\CoDeSys Sample Projects\netPLC PROFIBUS Simple IO
Short description	The project consist of a slot PLC card as PROFIBUS-DP Master and a PROFIBUS-DP slave device CBAB32-DPS. The PROFIBUS-DP slave device has two bytes input process data (occupying %IB0 and %IB1) and two output process data (occupying %QB0 and %QB1).
PLC program	The PLC program increments the %QB0 each cycle to act as a counter. And the PLC program copies the input process data byte %IB1 to output process data byte %QB1.

Table 26: Example – NPLC-C100-DP PROFIBUS Simple IO

13.2.5 NPLC-C100-DP with Host IOs

Project name	NPLC-C100 Host IOs.project
Folder	Supplements & Examples\CoDeSys Sample Projects\NPLC-C100-DP with Host IOs
Short description	The project consist of a slot PLC card as PROFIBUS-DP Master and a PROFIBUS-DP slave device CBAB32-DPS. The PROFIBUS-DP slave device has two bytes input process data (occupying %IB0 and %IB1) and two output process data (occupying %QB0 and %QB1).
PLC program	The PLC program increments the %QB0 each cycle to act as a counter. And the PLC program copies the input process data byte %IB1 to output process data byte %QB1. The addresses for host I/O data start from IB2 and %QB2. %QB2 is incremented by 1 in each PLC cycle. %IB513 is copied to %QB513. The sum of the values of %IB5 and %IB0 are written to %QB5.

Table 27: Example – NPLC-C100-DP with Host IOs

© Hilscher, 2009-2014

13.3 Information on Utilized Capacity

Information about required and available memory is displayed in the **Messages** area (scroll down to the bottom of **Messages** area) when you build code for the netPLC component (menu **Build** > **Generate Code**).

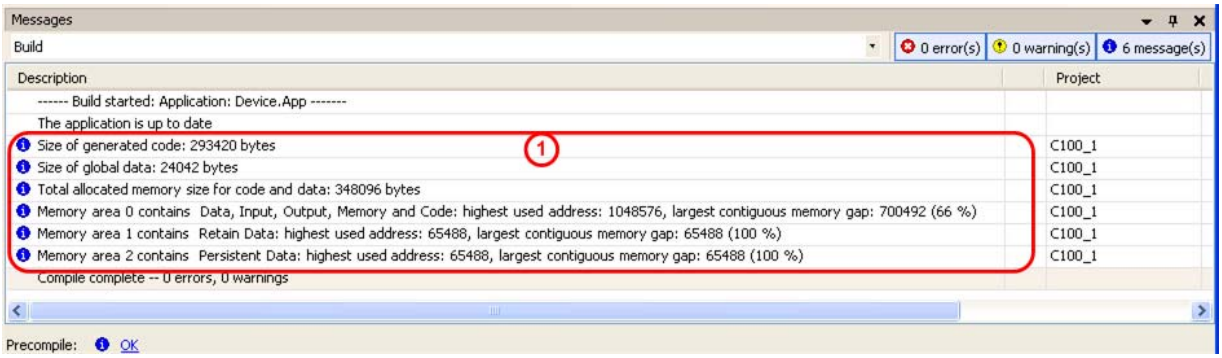
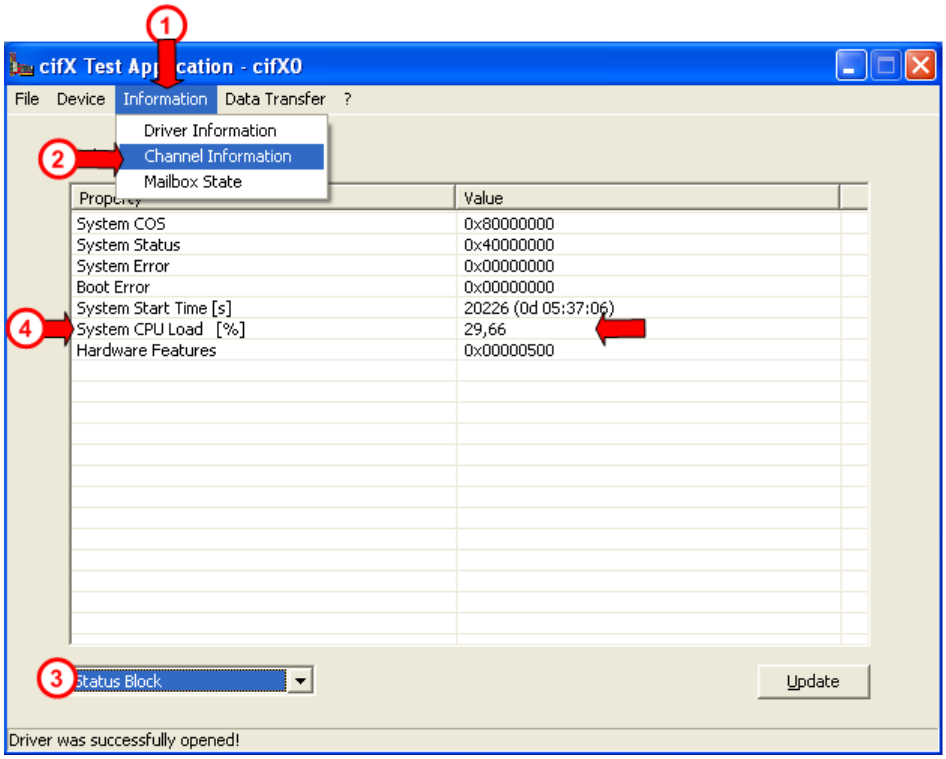

Messages
Build
0 error(s)
0 warning(s)
6 message(s)
Description
Project
------ Build started: Application: Device.App -------
The application is up to date
Size of generated code: 293420 bytes
Size of global data: 24042 bytes
Total allocated memory size for code and data: 348096 bytes
Memory area 0 contains Data, Input, Output, Memory and Code: highest used address: 1048576, largest contiguous memory gap: 700492 (66 %)
Memory area 1 contains Retain Data: highest used address: 65488, largest contiguous memory gap: 65488 (100 %)
Memory area 2 contains Persistent Data: highest used address: 65488, largest contiguous memory gap: 65488 (100 %)
Compile complete -- 0 errors, 0 warnings
C100_1
Precompile:
OK

Figure 54: Storage Requirement of Project

Information on storage requirement is indicated by position (1) in the picture above.

Information on the processor load caused by the project can be provided by the **cifX Test Application**, given that the netPLC component can be addressed via PCI bus or Ethernet.

- In the **Control Panel** of Windows, open **cifX Test Application**.
- In the **cifX Test Application**, use **Device > Open** menu to connect to the netPLC component.



© Hilscher, 2009-2014

- In the menu, choose **Information** ① **> Channel Information** ②.
- In the drop-down list located at the bottom of the window, select **Status Block** ③.

You can see the processor load in the **System CPU Load** line ④.

© Hilscher, 2009-2014

13.4 List of Figures

Figure 1: CoDeSys Main Window 16
Figure 2: New Project 17
Figure 3: Standard Project, Device Selection 18
Figure 4: Standard Project, Device Overview 18
Figure 5: NPLC-C100 Add Host I/O (1) 19
Figure 6: NPLC-C100 Add Host I/Os (2) 20
Figure 7: C100 Configure Host I/Os (1) 20
Figure 8: C100 Configure Host I/Os (2) 21
Figure 9: NPLC-M100 Add Local I/O (1) 22
Figure 10: NPLC-M100 Add Local I/Os (2) 23
Figure 11: M100 Configure Local I/Os 23
Figure 12: Select Fieldbus 26
Figure 13: PROFIBUS Master NetX Configuration 27
Figure 14: PROFIBUS Master DP Parameter Configuration 28
Figure 15: CANopen 2. Open Add Device Dialog 29
Figure 16: CANopen Manager 30
Figure 17: NetX_CANbus > NetX Configuration 31
Figure 18: NetX_CANbus > CANbus 31
Figure 19: CANopen_Manager > CANopen_Manager 32
Figure 20: CANopen > Add Device 33
Figure 21: DeviceNet Scanner 34
Figure 22: NetX_CANbus > NetX Configuration 35
Figure 23: NetX_CANbus > CANbus 35
Figure 24: Set Fieldbus Address of Device 36
Figure 25: Import Device Description File 37
Figure 26: Add Slave Device 38
Figure 27: Select Slave Device 39
Figure 28: Added Slave Device 39
Figure 29: Ascertain I/O Addresses of Slave 40
Figure 30: Slave Device Added 41
Figure 31: Editor Symbols 45
Figure 32: Editor Symbols (2) 45
Figure 33: Editor Symbols Access Rights 46
Figure 34: Messages After Building Process 47
Figure 35: Communication Settings 48
Figure 36: Set Active Path 50
Figure 37: Path Set Active 50
Figure 38: Messages Building Process 52
Figure 39: PLC Program Started 54
Figure 40: OPC Server Configuration 57
Figure 41: Host I/O Interface 62
Figure 42: Add Library 63
Figure 43: Add Library 64
Figure 44: Input Assistant - Module Calls 65
Figure 45: System Function StatusOfBattery 66
Figure 46: System function StatusOfExt24VDC 68
Figure 47: System Function StatusOfPCIPower 69
Figure 48: Bus Diagnostic (1) PROFIBUS Example 73
Figure 49: Bus Diagnostics (2) PROFIBUS Example 73
Figure 50: Bus Diagnostic (3) Slave I/O Data 74
Figure 51: Object reference not set… 75
Figure 52: netPLC CoDeSys Server Tray Icon 83
Figure 53: netPLC CoDeSys Server Window 84
Figure 54: Storage Requirement of Project 89

© Hilscher, 2009-2014

13.5 List of Tables

Table 1: List of Revisions 5
Table 2: Reference to netPLC Components (Hardware) 7
Table 3: Reference to Firmware 7
Table 4: Reference to Driver 7
Table 5: Reference to Server 7
Table 6: Reference to Software 7
Table 7: Documentations netPLC CoDeSys 8
Table 8: System Functions – HilPlcSysFunctions 64
Table 9: System Function StatusOfBattery 66
Table 10: System Function StatusOfBattery, Hardware without AD Converter 67
Table 11: System Function StatusOfExt24VDC 68
Table 12: System Function StatusOfPCIPower 69
Table 13: RUN/STOP Switch 70
Table 14: LED-Benennung in der Gerätezeichnung und LED-Bedeutung und -Benennung nach Protokoll (Feldbus-Systeme) 76
Table 15: System LED 76
Table 16: LEDs DeviceNet Master 77
Table 17: LEDs PROFIBUS DP Master 78
Table 18: Error Codes Functions HilPlcSysFunctions_netPLC 79
Table 19: Technical Data CANopen Master Protocol 80
Table 20: Technical Data DeviceNet Master Protocol 81
Table 21: Technical Data PROFIBUS DP Master Protocol 82
Table 22: netPLC CoDeSys Server Windows – Items 84
Table 23: Example – netPLC System Function 86
Table 24: Example – netPLC Open Process Control 87
Table 25: Example – netPLC Real Time Clock 87
Table 26: Example – NPLC-C100-DP PROFIBUS Simple IO 88
Table 27: Example – NPLC-C100-DP with Host IOs 88

© Hilscher, 2009-2014

13.6 Contacts

Headquarters

Germany
Hilscher Gesellschaft für
Systemautomation mbH
Rheinstrasse 15
65795 Hattersheim
Phone: +49 (0) 6190 9907-0
Fax: +49 (0) 6190 9907-50
E-Mail: info@hilscher.com

Support
Phone: +49 (0) 6190 9907-99
E-Mail: de.support@hilscher.com

Subsidiaries

China
Hilscher Systemautomation (Shanghai) Co. Ltd.
200010 Shanghai
Phone: +86 (0) 21-6355-5161
E-Mail: info@hilscher.cn

Support
Phone: +86 (0) 21-6355-5161
E-Mail: cn.support@hilscher.com

France
Hilscher France S.a.r.l.
69500 Bron
Phone: +33 (0) 4 72 37 98 40
E-Mail: info@hilscher.fr

Support
Phone: +33 (0) 4 72 37 98 40
E-Mail: fr.support@hilscher.com

India
Hilscher India Pvt. Ltd.
New Delhi - 110 065
Phone: +91 11 43055431
E-Mail: info@hilscher.in

Italy
Hilscher Italia S.r.l.
20090 Vimodrone (MI)
Phone: +39 02 25007068
E-Mail: info@hilscher.it

Support
Phone: +39 02 25007068
E-Mail: it.support@hilscher.com

Japan
Hilscher Japan KK
Tokyo, 160-0022
Phone: +81 (0) 3-5362-0521
E-Mail: info@hilscher.jp

Support
Phone: +81 (0) 3-5362-0521
E-Mail: jp.support@hilscher.com

Korea
Hilscher Korea Inc.
Suwon, Gyeonggi, 443-734
Phone: +82 (0) 31-695-5515
E-Mail: info@hilscher.kr

Switzerland
Hilscher Swiss GmbH
4500 Solothurn
Phone: +41 (0) 32 623 6633
E-Mail: info@hilscher.ch

Support
Phone: +49 (0) 6190 9907-99
E-Mail: ch.support@hilscher.com

USA
Hilscher North America, Inc.
Lisle, IL 60532
Phone: +1 630-505-5301
E-Mail: info@hilscher.us

Support
Phone: +1 630-505-5301
E-Mail: us.support@hilscher.com

© Hilscher, 2009-2014